ATTITUDE CONTROL AND MULTIMEDIA REPRESENTATION OF AIR FORCE INSTITUTE OF TECHNOLOGY'S (AFIT'S) SIMULATION SATELLITE (SIMSAT)

THESIS

Joseph M. Fulton, Captain, USAF

AFIT/GA/ENY/00M-01

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/GA/ENY/00M-01

ATTITUDE CONTROL AND MULTIMEDIA REPRESENTATION OF AIR FORCE INSTITUTE OF TECHNOLOGY'S (AFIT'S) SIMULATION SATELLITE (*SIMSAT*)

THESIS

Presented to the Faculty of the Graduate School of Engineering and Management
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Astronautical Engineering

Joseph M. Fulton, B.S., M.B.A.
Captain, USAF

March, 2000

AFIT/GA/ENY/00M-01

ATTITUDE CONTROL AND MULTIMEDIA REPRESENTATION OF AIR FORCE INSTITUTE OF TECHNOLOGY'S (AFIT'S) SIMULATION SATELLITE (*SIMSAT*)

Joseph M. Fulton, B.S., M.B.A.
Captain, USAF

Approved:

07 Mar 00

Captain Gregory S. Agnes Ph.D. — Date
Thesis Advisor

06 MAR 00

Dr. Bradley S. Liebst Ph.D. — Date
Committee Member

6 MAR 00

Lieutenant Colonel David R. Jacques Ph.D. — Date
Committee Member

Acknowledgements

A project as large as building AFIT's simulation satellite (*SIMSAT*) was completed with the combined support of several people. It was these individuals who provided the necessary resources, experience, direction, and motivation. First, six individuals deserve recognition for the work that came before this project. Capt James Colebank, Capt Robert Jones, Capt George Nagy, Capt Randall Pollak, 1Lt Donald Mannebach, and Mr. Mike Hanke began the work on *SIMSAT* as the initial design team. These individuals' vision and systems expertise made it possible to begin construction with only minimal modifications.

Mr. Jay Anderson, Mr. Bob Bacon, and Mr. Andy Pitts ordered needed parts, provided excellent working conditions, and offered superior insight into design and power requirements. Jason Edem, a summer intern, worked along side me building *SIMSAT* from the ground up. His desire to see a finished product resulted in the successful construction of the satellite in less than three months.

Finally, I would like to thank the most important people who aided me during this arduous task, my family. Raising a new born is no easy task and my wife Maryann did this with little complaint. She may think that she had no impact on this project but without her support none of this would have been possible. Thank you, my love. You, Jenna, and Emily are the reasons why I work so hard.

Joseph M. Fulton

Table of Contents

List of Figures

Figure Page

Figure Page

Figure Page

Figure Page

List of Tables

Table Page

AFIT/GA/ENY/00M-01

Abstract

This document describes the systematic construction of the AFIT-sponsored program to develop a laboratory-based satellite simulator. The simulation satellite (*SIMSAT*) system will provide a useful tool for resident staff while teaching attitude control concepts. A brief overview of attitude control theory is provided as well as a discussion of the benefits of multimedia use in education. A detailed discussion of the satellite's components allows the reader to become familiar with each piece of *SIMSAT*. Software control models are provided as well as a multimedia lesson plan on satellite attitude control. Also included in this document are potential experimental uses in the areas of attitude control, precision pointing, and vibration suppression as well as continued modification of the multimedia presentation capabilities.

ATTITUDE CONTROL AND MULTIMEDIA REPRESENTATION OF AIR FORCE INSTITUTE OF TECHNOLOGY'S (AFIT'S) SIMULATION SATELLITE (*SIMSAT*)

I. Introduction

1.1 Background

The last three decades have seen an increased use of space as a resource. Individuals use the resources of space daily without ever realizing it. Satellites provide information for up-to-date weather reports, communications, and television programs. Space continues to play an important role in U.S. military operations as well–missile warning, global communications, and global positioning systems (GPS). Because of the value of space, the USAF has begun to move its focus from operating as an air force to becoming the premier air and space force by the year 2025 . Accordingly, the Air Force Institute of Technology (AFIT) develops a curriculum and conducts research in both space operations and astronautical engineering. However, most of the research is completed using only computer simulations and stationary lab experiments[7].

In 1998, a systems design team tackled the task of designing a realistic satellite simulator to aid AFIT in its space research. The simulation satellite (*SIMSAT*), see Figure 1.1, provides AFIT with the capability to conduct practical experiments regarding attitude control theory, positioning sensors, and vibration control. *SIMSAT* is capable of fully rotating about its yaw and roll axes. Rotation about the pitch axis, however, is limited due to the air pedestal on which it rests. This air pedestal is used to simulate almost zero friction and prevents external torques from being applied to the system. Data cables attaching

SIMSAT to ground station computers will also apply external torques. It is because of this that wireless communication systems are used to transmit and receive telemetry data.

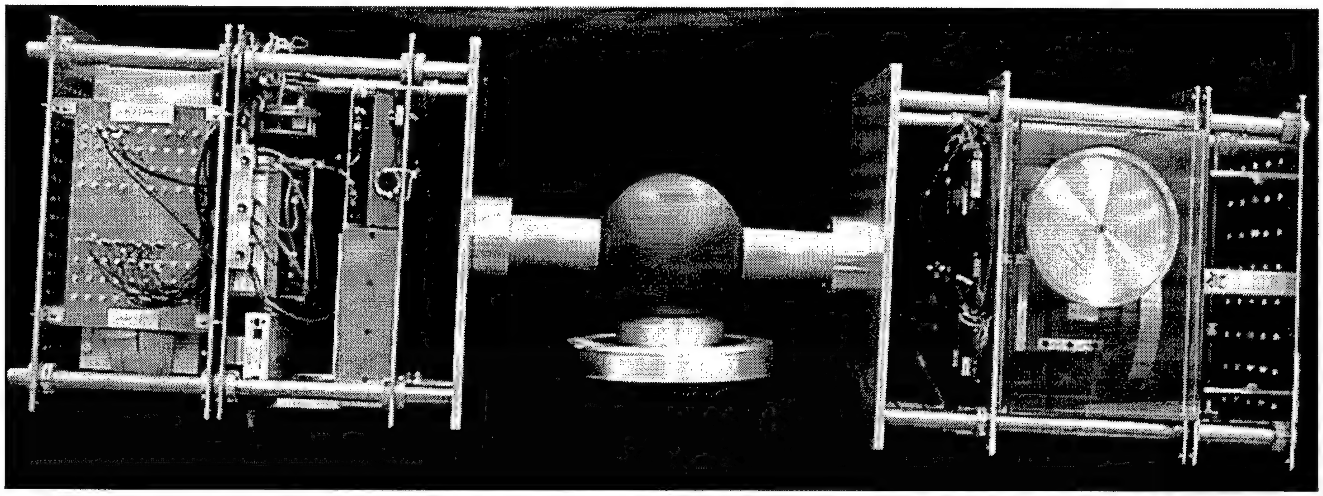


Figure 1.1 *SIMSAT*

SIMSAT would also improve classroom lectures by allowing students to visualize complex satellite attitude dynamics concepts and to gain hands-on experience with satellite control theory[13]. By adding to classic classroom instruction we hope to increase student interest in the course and to provide an invigorating learning environment. Studies have shown that students retain 25% of what they hear, 45% of what they see and hear, and almost 70% when they actively participate in the process[24]. The increase in retention percentages is a result of the student using more than one style of learning.

To better understand these multiple styles of learning we must first discuss how the mind operates. When individuals think of things, they do it associatively. When a person thinks of the word sun, many different images come to mind. This is the result of the linear (left brain) and spatial (right brain) aspects of the mind[2]. Rarely do individuals use only one side of the brain. Learned capabilities evolve from the combined efforts of both sides.

According to Gagne (1977) the human capacity for learning makes possible an infinite variety of behavioral patterns. He identified five varieties of learning capabilities[12]:

- intellectual skills (individual interacts with the environment using symbols)

- verbal information (states, tells, or writes a fact or set of events)
- cognitive strategies (manages own learning, remembering, and thinking)
- motor skills (executes organized movements)
- attitudes (mental state that influences choices)

Text, for example, focuses on the student's reading comprehension, or intellectual skills, and requires the reader to process the information in a uniform manner. Radio stimulates the same type of serial processing skills. Television, film, and face to face communication engage an individual's ability to synthesize information through sight and sound, requiring that person to simultaneously process several pieces of material.

Another researcher identifies three modes of representation for cognitive growth. The three ways in which somebody "knows" something is enactive (through doing it), ikonic (through a picture or image of it), and symbolic (through some symbolic means such as language)[5]. These three methods of learning something closely resemble a Chinese Proverb:

I hear and I forget. I see and I remember. I do and I understand.

While the traditional lecture is an effective way of presenting information to large groups in a short amount of time it is also the least effective method of promoting learning. When students are encouraged to take notes in class this adds another level of learning and reinforces the lecture material. However, to truly stimulate knowledge growth the students need to practice the taught skills and visualize the concepts presented in class[23].

1.1.1 Multimedia Defined. Throughout childhood, students are exposed to visual learning from television, video games, and the Internet[18]. Children grow up expecting dynamic presentations of information. This information stimulates their minds through use of color and sound. In 1991, most people throughout the world did not read about the Gulf War in the Middle East in the newspaper. CNN's televised reports brought the war into the living room. Television enabled Americans to watch live footage of the war and listen to government officials detail the military's operations. This is an example of how young people are receiving their information; creating a new generation which expects and responds to multisensory delivery systems[27].

More and more schools throughout the nation are taking steps to bring multisensory delivery systems into the classroom. By 1996 there were over 2,000 courses being offered on the World Wide Web and in the next couple of years it is estimated that over 75% of traditional US colleges and universities will use distance learning technologies in one or more traditional academic programs[31]. It is even possible to complete an undergraduate degree from start to finish over the internet. At the end of 1998, fifteen thousand elementary and middle schools had subscribed to an Internet network called Scholastic Network which provides Web-based classroom activities[37].

Some students are visual learners, auditory learners, or multisensory learners. Multimedia computer programs use text, pictures, sounds, animation, movie clips, and simulation to better reach students' levels of learning. Multimedia can mean several things:

> *Multimedia–the computer-based blending of graphics, sound, and video[15].*
>
> *Multimedia is commonly defined as the combination of text, graphics, audio, video and animation on a computer[11].*
>
> *Multimedia technology brings together video, graphics, animation, text, and sound in a single computer-controlled presentation[21].*

Generally, multimedia refers to two or more communications media under control of a computer. The student is able to interact with the material being taught instead of simply listening to a lecture.

1.1.2 Limitations of Multimedia. No configuration of screens, circuits, and discs can truly rival an expert teacher in giving and receiving information. However, does every student in America have the luxury of learning under an "expert" teacher? Instructors must address the limitations of including multimedia in the classroom before implementing attempts to reorganize educational practices. The issues of high costs, impacts on the instructor, cultural bias, and student loss of interaction skills with the world around them need to be considered by an institution if it is to pursue developing an educational environment capable of utilizing the benefits multimedia has to offer.

The high costs of multimedia are calculated in both time and money. The authoring process and the actual programming requires a substantial number of hours to complete. The author has to gather photos and video clips included in the program. Text is then integrated using a software package. Careful consideration to presentation style and content is critical to develop a product that is both visually stimulating and educationally beneficial to the user[18]. Authoring software packages are expensive. Quality educational packages tend to range from $500 to over $1,500. The hardware used to run and display multimedia projects also has a financial impact. The institution must decide how many computers to purchase, if the presentation of material will take place on an individual monitor or some type of large screen projector, and how much money should be spent on providing an adequate room where the learning will take place.

The expenses of securing multimedia material and a suitable learning environment are not the only costs involved. Teachers will need time, instruction, and support to achieve the level of expertise needed to employ the technology in their classrooms[36]. Time required to "teach the teacher" may be costly if the instructor lacks certain needed skills. According to Parris Powers, a chemistry instructor at Volunteer State Community College, teachers must have certain competencies to effectively incorporate multimedia material into their lectures[28]. These competencies are interpersonal communication skills, planning skills, collaboration and teamwork abilities, English proficiency, writing skills, organizational skills, feedback skills, knowledge of the multimedia education field, basic technology knowledge, and technology access capabilities. The first seven are essential to any instructor. The last three will allow an instructor to fully access students' various levels of learning. Some instructors may not have sufficient inservice time to learn how to use multimedia programs while others may not even be aware of the capabilities these programs offer[25]. Misuse of multimedia products will lead to ineffective teaching practices. Although many instructors routinely evaluate the progress of their students, the fear is some will give up this responsibility to the computer and no longer measure student development[16].

Cultural bias in the educational material is another issue that institutions need to consider. The obvious example of cultural bias in multimedia education is a student's lack of computer experience. The student's background is a factor since it has been shown that both poor and minority students had less access to computers at home and in school compared to middle-class Caucasian students[30]. This lack of experience may result in an increased level of apprehension when using the computer in school and may also lower a student's motivation to learn.

The subtle form of cultural bias is in the program itself. C. A. Bowers stresses that technology reduces awareness of the perspective adapted by the person who organizes the presented information[4]. This suggests that the software may reflect any bias feelings of the programmer. Some forms of this are gender, racial, religious, and philosophical in nature. M. W. Lee suggests that learning styles typical of white males form the basis of educational norms in America and that these styles are the most rewarded[17]. The learning styles which Lee refers to are field independence and analytical style. A large portion of students have difficulty thinking analytically. Instead, students use relational methods while studying.

The final limiting factor of multimedia concerns itself with students losing interaction skills between themselves and the world around them. Richard P. Lookatch, an educational psychologist with the Agency for Instructional Technology's Instructional Design Unit, speaks strongly on this issue.

> *Multimedia provides the opportunity to interact with the images behind a glass screen. The looming danger is that it replaces interaction with each other and the environment[19].*

He believes that multimedia education greatly reduces human interactions and hands-on experience since they are watching video clips of experiments instead of performing these tasks for themselves. Another article supports Lookatch's beliefs.

> *... a number of researchers have concluded that the cognitive processes most necessary for deeper level understanding and the implanting of information into memory occur only through dialogue and interaction with other people[14].*

Just as the telephone restricts personal communication by eliminating body language, the computer deprives young children the opportunity to interact socially with other children, a skill which is vital for an individual to function in an ever-growing society.

1.1.3 Benefits of Multimedia.

Educational institutions must research the limitations involved in using multimedia tools. However, these limitations should not deter institutions from the advantages multimedia offers. These advantages range from large sources of material to increased levels of student control on their learning. Increased quality of presentations contributes to improved student motivation and a richer lecture environment.

One advantage of multimedia materials over textbooks is databases can contain enormous amounts of information. Information that fills an entire bookcase is now stored on a simple computer disk. So much information is present that it would be impossible to cover all of the material in a single class.

> *Teachers who can proudly announce that their students "cover the entire textbook" each year without skipping any of the chapters will have a hard time doing the same thing if a CD replaces the text[20].*

It may even be extremely difficult for a teacher to present all of the material contained in multimedia software over a course of four school years.

With all of this information available, the student may feel overwhelmed unless the student maintains some control over the material viewed. Through the use of interactive multimedia programs, students are empowered with greater control over learning experiences. It is the computer's potential ability to respond to the learner which allows this interaction to take place[3]. The computer will keep track of the student's performance while the lessons are covered. If a student quickly understands a certain concept the computer will place less emphasis on the concept and focus more on an area where the student is having difficulty.

The key to interactivity is that the student is no longer passive in receiving information but is now taking an active role in the learning process. The student can make choices in how the material is presented. Would the student read the text on the screen or would he/she rather hear a recorded version of the text spoken to them? When viewing a video clip the student can slow down the images being displayed as well as see an instant replay. The choices even involve which order the material is presented. A student could see the results of a chemistry experiment before reading about the theory behind it. Student control also extends to enabling the student to open up a notepad and copy the text presented. This allows the student to write in additional comments and modify the copied text[8]. Personal interaction with the material can also take on other forms. Paul Merrill points out that with advancing technology voice recognition allows disabled children to activate programs by speech and text-to-speech software is useful for students who are visually impaired[22].

Another benefit of multimedia is an advanced level of lectures presented by instructors. One advantage is reduced lecture time. Instructors can write lecture material in presentation slides and then hand out those slides for students to take notes. No longer will the instructor have to transfer information to a chalkboard. In addition, instructors can incorporate visual aids such as animation and video with the slides. Fifield and Peifer showed when visual aids were incorporated into lectures students' recall and comprehension of the presented material increased[10]. With the use of stimulating visual aids, students are able to visualize real-world phenomena and eliminate any misconceptions. Parris Powers supports Fifield and Peifer's beliefs about the increased levels of comprehension being a direct result of students enjoying the new lectures. He showed that 97% of surveyed students preferred multimedia presentations over strict chalkboard leactures[28]. This enjoyment leads to increased motivation toward learning and helps build an improved learning environment.

Many researchers conducted studies concerning student reactions to multimedia educational presentations. Two of these researchers were H. Weiner and H. Pence. Weiner conducted a study to measure sixth grade middle school students' abilities on achievement tests in social studies after using computer based instruction. The study showed that individual scores on recalling information and utilizing geography skills increased an average of 40%[33]. Pence's study was centered on 100 college general chemistry students. He used a Powerpoint multimedia program along with lecture, overheads, and handouts to teach his class. Students commented the information seemed dynamic rather than static and that it was a better way of presenting the material rather than just using overheads[26].

The literature review found no documentation stating multimedia educational presentations had a negative effect on student motivation or inhibited effective learning. An important item to remember, however, is that it is not the machine that motivates. Researchers need to discover ways for the instructor to play an active role in student learning while reaping the many benefits multimedia materials are shown to offer. If researchers fail, the technology will make it easier for bad things to happen more efficiently.

1.2 Problem Statement

AFIT's 1998 Space Operations Systems Design team accepted the task of developing the initial design of *SIMSAT* and ordered the needed components. The next step involved assembling the components into an operational experimental satellite. The problem statement is as follows:

> *Construct* SIMSAT *and fully integrate system software to provide a testbed for further AFIT satellite research while at the same time providing a multimedia educational tool capable of demonstrating complex satellite attitude control theory.*

1.3 Objectives

The objectives of this research follow from the problem statement. In order of importance, this research will:

1. Assemble *SIMSAT* and determine design modifications
2. Determine the mathematical model of *SIMSAT*'s components
3. Integrate computer software to link *SIMSAT*'s on-board computer with command and control software
4. Develop a multimedia lesson plan on satellite dynamics concepts

This researcher accomplished the above objectives as well as detailed future research opportunities.

1.4 Document Overview

This report documents the assembly and operation of AFIT's new experimental simulation satellite. Chapter II provides an abbreviated version of attitude control theory taught at AFIT. The various structural and functional components of *SIMSAT* are discussed in Chapter III. This chapter also includes brief descriptions of the software programs used to operate *SIMSAT*. Experimental setup and results are provided in Chapter IV. Information pertaining to the multimedia lesson plan and laboratory exercise is also located in Chapter IV. Chapter V concludes the report and summarizes the entire research effort by providing information on continuing research topics.

II. Attitude Control Theory

2.1 Overview

A rigid body has six degrees of freedom. Three degrees identify the translation of an object and the other three provide information to its rotational orientation. For experimental purposes *SIMSAT* is considered a rigid body. In the lab, *SIMSAT* rests on its air bearing assembly as shown in Figure 2.1. For information on the air bearing assembly, reference Section 3.2.1. The spherical rotor floats on a thin film of compressed air and allows *SIMSAT* to rotate about all three axes without any translation.

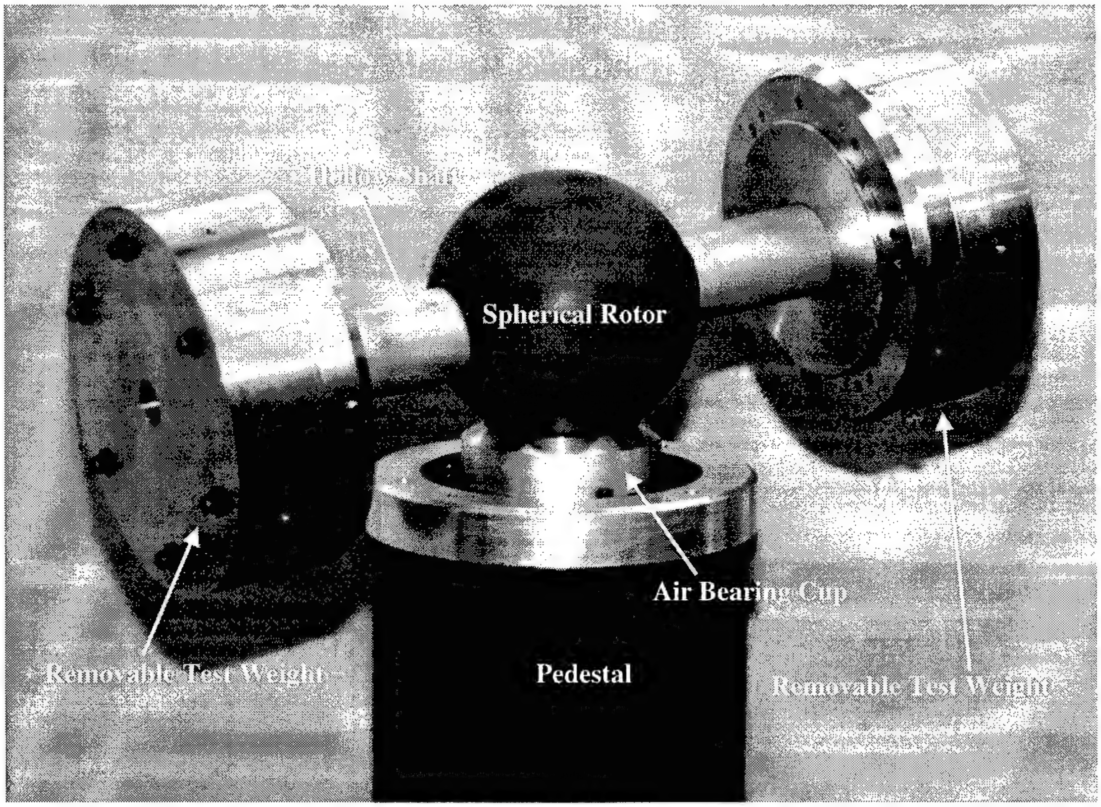


Figure 2.1 Air Bearing Assembly

This chapter provides an abbreviated version of attitude control theory taught at AFIT. The theory of rotational kinematics is only covered since *SIMSAT*'s translational motion is absent. *SIMSAT* is treated as a rigid body for simulation purposes and therefore a discussion of rigid body dynamics is included. The chapter concludes with attitude control concepts followed by a section on momentum wheel control theory.

2.2 Rotational Kinematics

When studying the equations of motion of an object one must understand both kinetics and kinematics. Kinetics is the study of the forces acting on a body while kinematics describes the motion of that body. This section will concern itself with the theory of rotational kinematics. Kinetics will be discussed in Section 2.3. "The subject of rotational kinematics is somewhat mathematical in nature because it does not involve any forces associated with motion" [34]. The three mathematical approaches to identifying rotational motion are direction cosines, Euler angles, and quaternions.

2.2.1 Direction Cosines.

The first mathematical method of describing rotational kinematics is direction cosines. In a rigid body there exists a body reference frame A. This frame consists of a right-hand set of three orthogonal unit vectors $\{\vec{a_1}, \vec{a_2}, \vec{a_3}\}$. Another reference frame B will contain a different set of three right-handed orthogonal unit vectors $\{\vec{b}_1, \vec{b}_2, \vec{b_3}\}$. The dot product law is used to describe one reference frame in terms of the other.

$$\vec{a} \cdot \vec{b} = |a||b| \cos \angle \vec{a}\vec{b} \tag{2.1}$$

Using the dot product B can now be expressed in terms of A as follows:

$$\begin{aligned}
\vec{b_1} &= C_{11}\vec{a_1} + C_{12}\vec{a_2} + C_{13}\vec{a_3} \\
\vec{b_2} &= C_{21}\vec{a_1} + C_{22}\vec{a_2} + C_{23}\vec{a_3} \\
\vec{b_3} &= C_{31}\vec{a_1} + C_{32}\vec{a_2} + C_{33}\vec{a_3}
\end{aligned} \tag{2.2}$$

where $C_{ij} \equiv \vec{b_i} \cdot \vec{a_j}$ is the cosine of the angle between $\vec{b_i}$ and $\vec{a_j}$ and C_{ij} is called the direction cosine.

An equivalent way of expressing the directional cosines is in matrix form.

$$\begin{pmatrix} \vec{b_1} \\ \vec{b_2} \\ \vec{b_3} \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{pmatrix} \begin{pmatrix} \vec{a_1} \\ \vec{a_2} \\ \vec{a_3} \end{pmatrix} \tag{2.3}$$

The direction cosine matrix is also called the rotation matrix or the coordinate transformation matrix [1].

The kinematical equations of Poisson describe the functional relationships of the direction cosines and their rates [6].

$$\begin{aligned}
\dot{C}_{11} &= C_{12}\omega_3 - C_{13}\omega_2 \\
\dot{C}_{12} &= C_{13}\omega_1 - C_{11}\omega_3 \\
\dot{C}_{13} &= C_{11}\omega_2 - C_{12}\omega_1 \\
\dot{C}_{21} &= C_{22}\omega_3 - C_{23}\omega_2 \\
\dot{C}_{22} &= C_{23}\omega_1 - C_{21}\omega_3 \\
\dot{C}_{23} &= C_{21}\omega_2 - C_{22}\omega_1 \\
\dot{C}_{31} &= C_{32}\omega_3 - C_{33}\omega_2 \\
\dot{C}_{32} &= C_{33}\omega_1 - C_{31}\omega_3 \\
\dot{C}_{33} &= C_{31}\omega_2 - C_{32}\omega_1
\end{aligned} \tag{2.4}$$

The direction cosines are easy to calculate but do require integrating 9 equations to solve rotational kinematics. Also, the direction cosines are not intuitive since the values are not expressed as angles.

2.2.2 Euler Angles.

A more intuitive way of looking at rotational kinematics is with Euler angles. These angles are easier to visualize than direction cosines. A perfect example of Euler angles is the yaw, pitch, and roll of an aircraft. Euler angles involve rotating about the three axes of the body as shown in Figure 2.2.

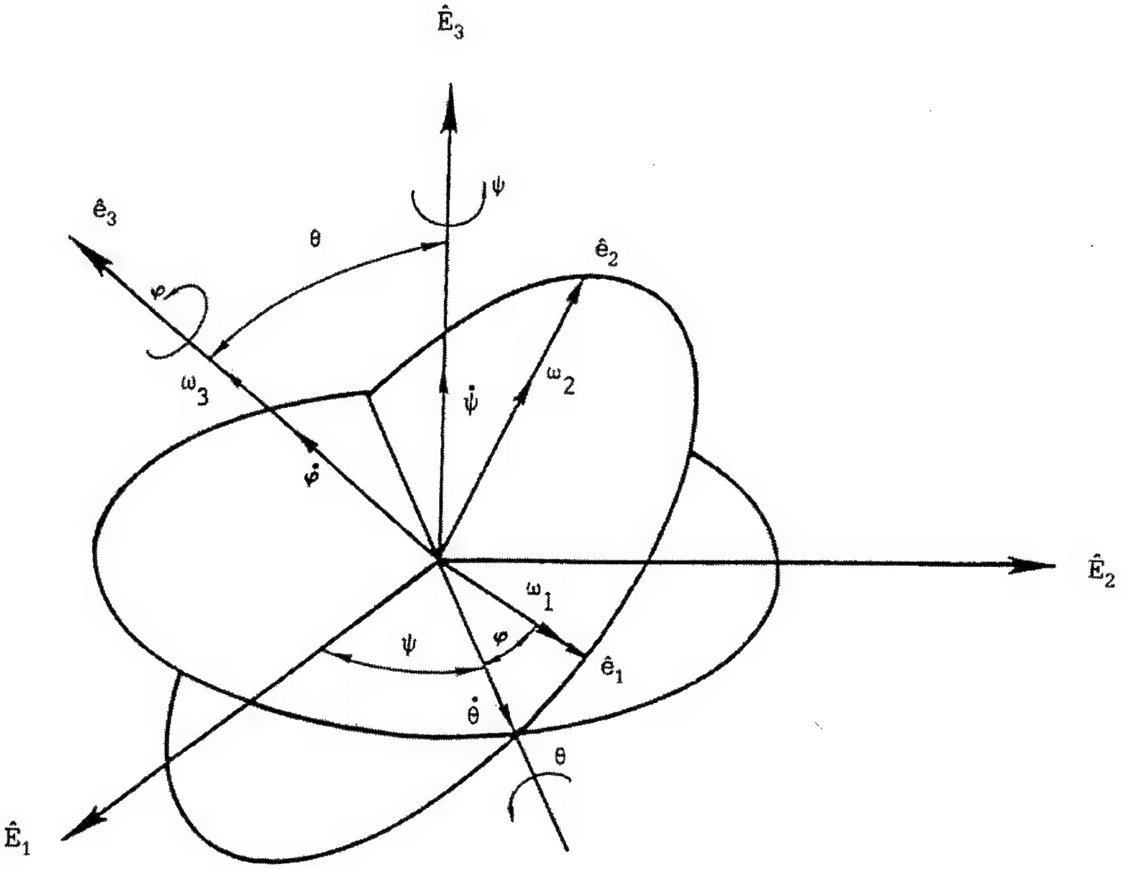


Figure 2.2 Classical Euler rotations of a rigid body [6]

For each rotation about an axis, a rotational matrix is calculated. Consider a rotation of a body axis $\hat{e}$ with respect to a reference frame $\hat{E}$ as shown in Figure 2.3. The components of $\hat{E}$ along the $\hat{e}$ directions are calculated as

$$\begin{aligned} \hat{e}_1 &= \hat{E}_1 \\ \hat{e}_2 &= \hat{E}_2 \cos\theta_1 + \hat{E}_3 \sin\theta_1 \\ \hat{e}_3 &= -\hat{E}_2 \sin\theta_1 + \hat{E}_3 \cos\theta_1 \end{aligned} \tag{2.5}$$

or in matrix form as

$$\begin{pmatrix} \hat{e_1} \\ \hat{e_2} \\ \hat{e_3} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c\theta_1 & s\theta_1 \\ 0 & -s\theta_1 & c\theta_1 \end{pmatrix} \begin{pmatrix} \hat{E}_1 \\ \hat{E}_2 \\ \hat{E}_3 \end{pmatrix} \tag{2.6}$$

where $c\theta$ and $s\theta$ represent $\cos\theta$ and $\sin\theta$.

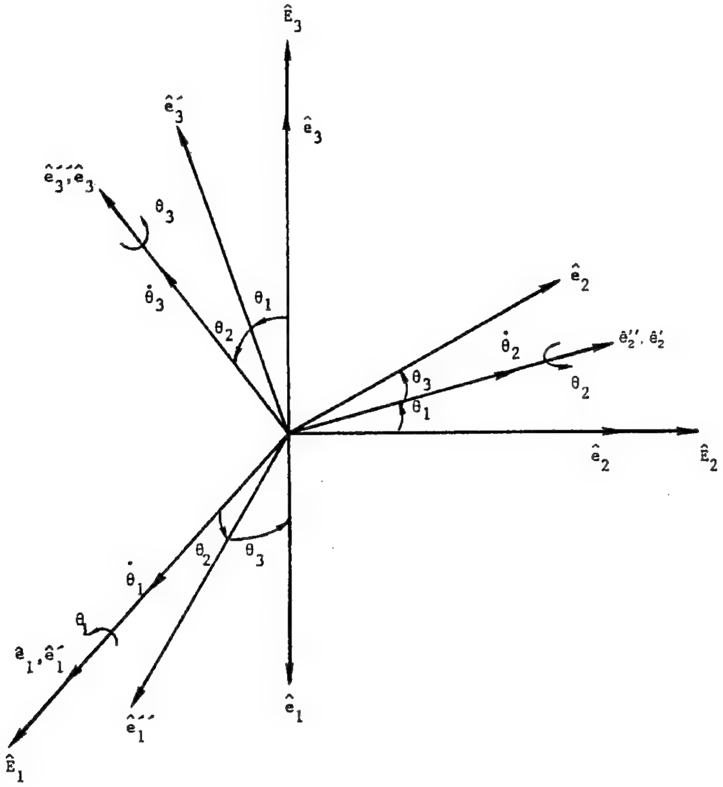


Figure 2.3 Sequential orthogonal rotations of the $\hat{e}$ reference frame about the $\hat{E}$ reference frame [6]

A rotation about each axis can be represented as

$$\begin{pmatrix} \hat{e}_1' \\ \hat{e}_2' \\ \hat{e}_3' \end{pmatrix} = R(\theta_1) \begin{pmatrix} \hat{E}_1 \\ \hat{E}_1 \\ \hat{E}_1 \end{pmatrix} \tag{2.7}$$

$$\begin{pmatrix} \hat{e}_1'' \\ \hat{e}_2'' \\ \hat{e}_3'' \end{pmatrix} = R(\theta_2) \begin{pmatrix} \hat{e}_1' \\ \hat{e}_2' \\ \hat{e}_3' \end{pmatrix} \tag{2.8}$$

$$\begin{pmatrix} \hat{e}_1 \\ \hat{e}_2 \\ \hat{e}_3 \end{pmatrix} = R(\theta_3) \begin{pmatrix} \hat{e}_1'' \\ \hat{e}_2'' \\ \hat{e}_3'' \end{pmatrix} \tag{2.9}$$

where

$$R(\theta_1) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c\theta_1 & s\theta_1 \\ 0 & -s\theta_1 & c\theta_1 \end{pmatrix} \tag{2.10}$$

$$R(\theta_2) = \begin{pmatrix} c\theta_2 & 0 & -s\theta_2 \\ 0 & 1 & 0 \\ s\theta_2 & 0 & c\theta_2 \end{pmatrix} \tag{2.11}$$

$$R(\theta_3) = \begin{pmatrix} c\theta_3 & s\theta_3 & 0 \\ -s\theta_3 & c\theta_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.12}$$

Referring again to Figure 2.2, assume the body is undergoing a rotation first about the $\hat{e}_1$ axis followed by a rotation about $\hat{e}_2$ and then $\hat{e}_3$. The body reference frame is expressed in terms of the fixed reference frame as follows:

$$\begin{pmatrix} \hat{e}_1 \\ \hat{e}_2 \\ \hat{e}_3 \end{pmatrix} = R(\theta_3)R(\theta_2)R(\theta_1) \begin{pmatrix} \hat{E}_1 \\ \hat{E}_2 \\ \hat{E}_3 \end{pmatrix} \tag{2.13}$$

or

$$
\begin{aligned}
R_{123} &= R(\theta_3)R(\theta_2)R(\theta_1) \\
&= \begin{pmatrix} c\theta_2 c\theta_3 & c\theta_3 s\theta_1 s\theta_2 + c\theta_1 s\theta_3 & -c\theta_1 c\theta_3 s\theta_2 + s\theta_1 s\theta_3 \\ -c\theta_2 s\theta_3 & -s\theta_1 s\theta_2 s\theta_3 + c\theta_1 c\theta_3 & c\theta_1 s\theta_2 s\theta_3 + s\theta_1 c\theta_3 \\ s\theta_2 & -c\theta_2 s\theta_1 & c\theta_1 c\theta_2 \end{pmatrix} \qquad (2.14)
\end{aligned}
$$

The angular velocities for this rotation sequence are

$$
\begin{aligned}
\omega_1 &= \dot{\theta}\cos\phi + \dot{\psi}\sin\theta\sin\phi \\
\omega_2 &= \dot{\psi}\sin\theta\cos\phi - \dot{\theta}\sin\phi \\
\omega_3 &= \dot{\phi} + \dot{\psi}\cos\theta \qquad (2.15)
\end{aligned}
$$

Solving for $\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$ yields

$$
\begin{pmatrix} \dot{\psi} \\ \dot{\phi} \\ \dot{\theta} \end{pmatrix} = \frac{1}{\sin\theta} \begin{pmatrix} \sin\phi & \cos\phi & 0 \\ -\sin\phi\cos\theta & -\cos\phi\cos\theta & \sin\theta \\ \cos\phi\sin\theta & -\sin\phi\sin\theta & 0 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} \qquad (2.16)
$$

These three equations appear easy to integrate but trigonometric functions require greater computational time than addition and multiplication. Although Euler angles have fewer equations to integrate than direction cosines, a larger amount of processing time is used to calculate the angular velocities. With *SIMSAT*, computational time is not a concern and the simulation model uses Euler angle calculations. The only concern is the singularity in Eq. 2.16 when θ equals 0°. *SIMSAT*'s pitch angle is measured from the horizontal axis. Thus, θ has an initial condition set at 0°. Since the air bearing assembly limits *SIMSAT*'s motion to ±25° of pitch from the horizontal plane, a different rotation sequence is used.

To avoid the singularity in pitch, a roll-pitch-yaw (1-3-2) rotation sequence is used. This rotation sequence yields

$$\begin{pmatrix} \dot{\psi} \\ \dot{\phi} \\ \dot{\theta} \end{pmatrix} = \frac{1}{\cos\theta} \begin{pmatrix} \cos\theta & -\cos\psi\sin\theta & \sin\psi\sin\theta \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi\cos\theta & \cos\psi\cos\theta \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} \tag{2.17}$$

Now the singularity occurs when the pitch angle equals 90°.

2.2.3 Quaternions.

The third mathematical approach to identifying rotational motion is using quaternions. Instead of the trigonometric functions of Euler angles, quaternions use algebraic relations. This provides quicker computations and also eliminates the possibility of a singularity appearing. An Euler axis, also known as a principal axis, is an axis that is fixed in the body frame and is stationary in the inertial frame. This axis is special because any combination of rigid body rotation is described as a single rotation about the Euler axis. The eigenaxis vector $\vec{e} = (e_1, e_2, e_3)$ is simply the direction cosines of the Euler axis relative to both the body and inertial reference frames.

The quaternions, Euler parameters, are then defined as

$$\begin{aligned} q_1 &= e_1 \sin(\theta/2) \\ q_2 &= e_2 \sin(\theta/2) \\ q_3 &= e_3 \sin(\theta/2) \\ q_4 &= \cos(\theta/2) \end{aligned} \tag{2.18}$$

where θ is the rotation angle about the Euler axis. After applying trigonometric identities the rotation matrix becomes

$$R = \begin{pmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2+q_3q_4) & 2(q_1q_3-q_2q_4) \\ 2(q_1q_2-q_3q_4) & 1-2(q_1^2+q_3^2) & 2(q_2q_3+q_1q_4) \\ 2(q_1q_3+q_2q_4) & 2(q_2q_3-q_1q_4) & 1-2(q_1^2+q_2^2) \end{pmatrix} \tag{2.19}$$

Substituting Eq. 2.19 into Eqs. 2.4 and solving for angular velocity we obtain

$$\omega_1 = 2(\dot{q}_1q_4 + \dot{q}_2q_3 - \dot{q}_3q_2 - \dot{q}_4q_1)$$

$$\begin{aligned}
\omega_2 &= 2(\dot{q}_2 q_4 + \dot{q}_3 q_1 - \dot{q}_1 q_3 - \dot{q}_4 q_2) \\
\omega_3 &= 2(\dot{q}_3 q_4 + \dot{q}_1 q_2 - \dot{q}_2 q_1 - \dot{q}_4 q_3) \qquad (2.20)
\end{aligned}$$

Applying a constraint equation such that

$$q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1 \qquad (2.21)$$

and differentiating it

$$0 = 2(\dot{q}_1 q_1 + \dot{q}_2 q_2 + \dot{q}_3 q_3 + \dot{q}_4 q_4) \qquad (2.22)$$

We can now combine Eqs. 2.20 and Eq. 2.22 into matrix form as follows:

$$\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ 0 \end{pmatrix} = 2 \begin{pmatrix} q_4 & q_3 & -q_2 & -q_1 \\ -q_3 & q_4 & q_1 & -q_2 \\ q_2 & -q_1 & q_4 & -q_3 \\ q_1 & q_2 & q_3 & q_4 \end{pmatrix} \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{pmatrix} \qquad (2.23)$$

Rearranging Eq. 2.23 results in the kinematic differential equations for quaternions as

$$\begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} q_4 & -q_3 & q_2 & q_1 \\ q_3 & q_4 & -q_1 & q_2 \\ -q_2 & q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ 0 \end{pmatrix} \qquad (2.24)$$

or

$$\begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 0 & \omega_3 & -\omega_2 & \omega_1 \\ -\omega_3 & 0 & \omega_1 & \omega_2 \\ \omega_2 & -\omega_1 & 0 & \omega_3 \\ -\omega_1 & -\omega_2 & -\omega_3 & 0 \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix} \qquad (2.25)$$

As a result of the fewer number of equations to integrate compared to direction cosines, and the absence of singularities and trigonometric functions, modern spacecraft orientation is now commonly described in terms of quaternions [34].

2.3 Rigid Body Dynamics

Kinematic differential equations are useful for integrating the motion of a satellite. An understanding of kinetics is required to study how this satellite reacts to forces acting upon it. This section will first discuss the properties of angular momentum, moments of inertia, and how moments will affect the equations of motion. A discussion on stability requirements for both linear and nonlinear systems is provided followed by an explanation of the attitude control systems used to stabilize spacecraft. The section concludes with a detailed discussion of the equations used in designing a momentum wheel controller.

2.3.1 Kinetics. Just as a moving body has translational momentum comprised of its velocity and mass, a rotating body will also have rotational momentum. This rotational momentum is referred to as angular momentum. Similar to its counterpart, angular momentum is comprised of angular velocity, ω, and inertia, I. Thus the total angular momentum is expressed as

$$\vec{H} = I\vec{\omega} \tag{2.26}$$

Both $\vec{H}$ and $\vec{\omega}$ are 3×1 vectors so I is a 3×3 matrix known as the inertia matrix.

A rigid body does not necessarily have to possess uniform density distribution. It's density can vary with position relative to the system's center of mass. The moment of inertia matrix contains all of the information of the mass distribution within a rigid body [35].

$$I = \begin{pmatrix} \int(y^2+z^2)dm & -\int xydm & -\int xzdm \\ -\int xydm & \int(x^2+z^2)dm & -\int yzdm \\ -\int xzdm & -\int yzdm & \int(x^2+y^2)dm \end{pmatrix} \tag{2.27}$$

This inertia matrix only needs to be recalculated if the mass distribution alters from its original configuration. With *SIMSAT*, the inertial properties will change when an experiment is added to the system.

The kinetic relation between moments and angular momentum is

$$M = \dot{H} \tag{2.28}$$

Using a body fixed reference frame Eq. 2.28 becomes

$$M = \frac{d}{dt}H + \omega \times H \tag{2.29}$$

Applying the matrix form of the cross product yields

$$M = I\dot{\omega} + \omega^x I\omega \tag{2.30}$$

where ω^x is a scew-symmetric matrix of the form

$$\omega^x = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \tag{2.31}$$

To further simplify the equations of motion, assume a principal axis frame. The inertia matrix becomes a diagonal matrix

$$I = \begin{pmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{pmatrix} \tag{2.32}$$

and Eq. 2.30 becomes

$$\begin{pmatrix} M_1 \\ M_2 \\ M_3 \end{pmatrix} = \begin{pmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{pmatrix} \begin{pmatrix} \dot{\omega}_1 \\ \dot{\omega}_1 \\ \dot{\omega}_1 \end{pmatrix} + \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \begin{pmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_1 \\ \omega_1 \end{pmatrix} \tag{2.33}$$

which reduces to Euler's rotational equations of motion for a rigid body.

$$\begin{aligned} M_1 &= I_1\dot{\omega}_1 - (I_2 - I_3)\omega_2\omega_3 \\ M_2 &= I_2\dot{\omega}_2 - (I_3 - I_1)\omega_1\omega_3 \\ M_3 &= I_3\dot{\omega}_3 - (I_1 - I_2)\omega_1\omega_2 \end{aligned} \tag{2.34}$$

These three equations are coupled, nonlinear ordinary differential equations. The rotational motion of a rigid body can be completely described when Euler's rotational equations of motion are combined with the kinematic differential equations from Section 2.2.

2.3.2 Stability. Stability is a major concern with satellite control theory. An unstable satellite may drift away from its operational orbital path, disrupt communications, or waste excessive fuel and reduce its mission effectiveness. An object near an equilibrium point is considered stable when small disturbances result in small changes. To understand stability about an equilibrium point consider a pendulum. Two equilibrium points exist. One point occurs when the pendulum is pointing straight down and the other point when the pendulum is pointing straight up. At both equilibrium points the pendulum will remain motionless if no external forces are applied. However, only one of these points are stable. A small disturbance applied to the pendulum while pointing down will cause it to oscillate slightly about its equilibrium point. This configuration is considered stable. A small disturbance applied to the pendulum while it is pointing up will cause a large change in its orientation. This point is unstable.

Current theory supports two concepts of stability. The first concept deals with Lagrange stability. Lagrange stability applies when small disturbances result in bounded changes. Liapunov stability occurs when an object resting at an equilibrium point experiences small disturbances and only small changes are detected. Now consider the pendulum example previously mentioned. When the pendulum is pointing downward it is considered to be both Liapunov and Lagrange stable. When the pendulum is pointing straight up, any small change will result in a drastic change in location. Thus, this point is Liapunov unstable. But the angle of the pendulum is bounded (i.e. it may oscillate slightly about the equilibrium position) so the system is considered Lagrange stable. From this it is apparent that a Liapunov stable system is also Lagrange stable but a Lagrange stable system is not necessarily Liapunov stable.

To test if a linear system is considered stable a concept from control theory is applied. "A linear system is said to be stable if and only if the roots of its characteristic equation have negative real parts" [6]. This is the same as saying that the real part of the eigenvalues of the system are only negative. A zero root is considered asymptotically unstable. A system is asymptotically stable if changes from an equilibrium point result in returning to the same equilibrium point.

Liapunov developed two theorems to determine if a nonlinear system is stable. The first theorem states that a nonlinear system is asymptotically stable in a region near an equilibrium point if and only if its linearized approximation is asymptotically stable. The second theorem states if a positive definite function can be found such that its derivative is negative definite the nonlinear system is stable [1]. This function is often referred to as the Liapunov function. For a more indepth discussion of stability and methods to determining a Liapunov function refer to [6], pages 113-126.

2.4 Attitude Control Systems

All stable spacecraft have some type of pointing requirement. Solar panel orientation and communication antenna positioning are two such requirements. Several methods are used to ensure these requirements are met. These methods can either be passive or active. Passive control systems are spin, dual spin, and gravity gradient control. Three axis control is an active control system which utilizes thrusters, momentum wheels, and/or control moment gyros.

2.4.1 Spin Control. Spin control is the simplest of control methods. It utilizes gyroscopic stability as the satellite spins about its major axis. A minor axis spinner is stable as long as active nutation control is applied to offset the effects of energy dissipation. The gyroscopic stability of the satellite creates a resistance to off-axial thrust. Pointing accuracy of a spinning satellite is approximately $\pm 0.1°$. One of the drawbacks of a spinning satellite is poor maneuverability resulting from high angular momentum. The other drawback is a spinning satellite using solar power will have low levels of power available since only half of the solar cells will be in direct sunlight at any time.

2.4.2 Dual-Spin Control. A dual-spinner uses the same techniques of a spin controlled spacecraft except a second platform is spun up. The platform and rotor

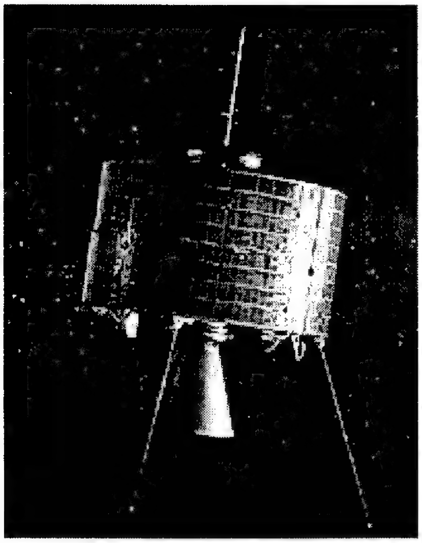


Figure 2.4 Syncom–A typical spin stabilized spacecraft

spin at different rates. A dual-spin controlled spacecraft will have all of the advantages and disadvantages of a spin controlled spacecraft with an additional advantage. While the rotor acts as if it is a pure spinner, the platform can rotate along with the rotation of the Earth. This allows scanning equipment placed on the platform to be fixed inertially.

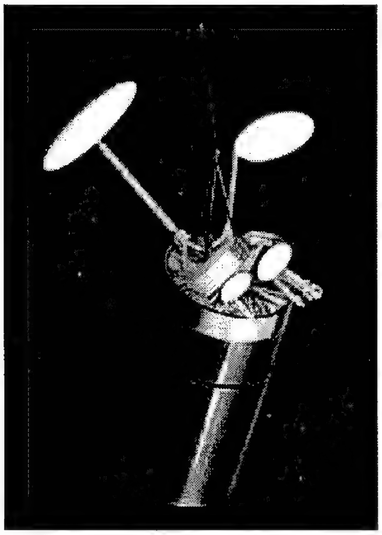


Figure 2.5 Intelsat VI–A typical dual-spin spacecraft.

2.4.3 ***Gravity Gradient Control.*** The center of gravity is often times not in the same location as the center of mass for a large object orbiting about a planet.

The differing locations will result in a torque applied to the object as a result of differing gravitational strength. A simple solution to overcome this applied torque in a gravity gradient field is to attach a cable to the satellite. A mass is attached to the end of the cable as shown in Figure 2.6, pointing either toward or away from the earth. The cable is slowly let out until a stable configuration is achieved. Although the design is low cost and simple to implement, its pointing accuracy is only 10 to 20 degrees. An active form of control, such as a momentum wheel, can increase this pointing accuracy to ±0.005°.

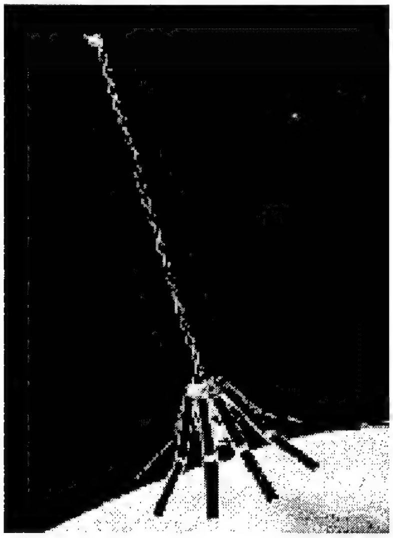


Figure 2.6 GEOSAT–A gravity gradient stabilized spacecraft.

2.4.4 Three-Axis Control. The previous examples were those of passive control in which a satellite is deployed and either spun up or a cable is extended and then left alone. Without any other control systems installed, an operator does not have the capability to actively control the satellite in response to catastrophic events. With a three-axis control system an operator possesses the capability of actively manipulating the orientation of a satellite. The system gets its name because each axis of the satellite is influenced by either thrusters, momentum wheels, or control moment gyros. Now, solar panels are aligned for maximum output of power and pointing accuracy is increased by an order of magnitude over basic spinning systems. Another advantage is the satellite can adapt to changing mission requirements through maneuvering. *SIMSAT* is a three-axis controlled satellite which uses momentum wheels. Applying cold gas thrusters or control moment gyros are recommended in Chapter V.

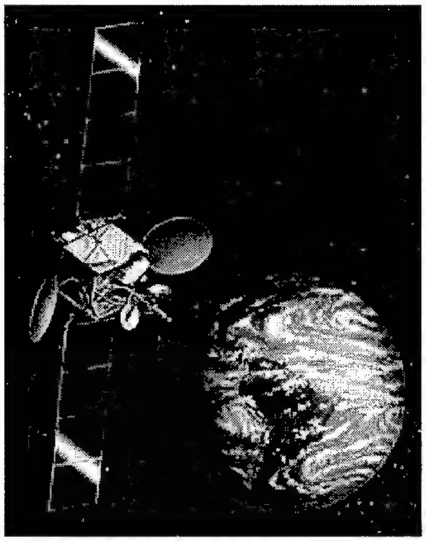


Figure 2.7 PanAmSat–A three-axis controlled spacecraft.

2.4.4.1 Thrusters.

Thrusters actively apply torques to a satellite to affect its orientation. Several types of thrusters are used currently. Cold gas jets contain pressurized propellant which is vented through a nozzle. Hydrazine monopropellant thrusters use a chemical reaction and eject the hot gas. Modern electrical thrusters, such as ion thrusters or pulsed plasma thrusters, eject ionized gas. Their low fuel consumption makes them ideal candidates for station keeping missions. However, these systems do require large power sources to operate. Pointing accuracy of a thruster control system is ±0.1°. A major disadvantage of a thruster control system is the consumption of fuel. Each thruster contains a limited supply of propellant. Once this fuel is depleted the satellite is no longer controllable.

2.4.4.2 Momentum Wheels.

Another method of actively controlling the stability of a spacecraft is momentum wheels. This system consists of a flywheel mounted either on each axis or two axes coupled together to provide control over the third. This control system is lighter and costs less when compared to a thruster system. Pointing accuracy is also increased to ±0.01°.

2.4.4.3 Control Moment Gyros.

A control moment gyro (CMG) is similar to a momentum wheel, except the wheel is now mounted in a gimbal. Another difference is when the satellite is not moving inertially, the CMG wheel is always spinning while the momentum wheel is not. Control of the spacecraft is achieved by slowly rotating the gimbal. The angular momentum components of the CMG change from one axis to another. The wheel spins about one axis and the gimbal rotates about a second axis. This results in a torque applied to the third axis. A single CMG does not provide control over all three axes. A second system is required to fully control all three axes and a third CMG is typically added to acquire full redundancy in controlling a spacecraft. Pointing accuracy is similar to those of momentum wheels.

2.5 Momentum Wheel Controller

Consider a momentum wheel aligned with a single axis. The magnitude of the angular momentum of the wheel is expressed as

$$H_w = J(\dot{\theta} + \Omega) \tag{2.35}$$

and the angular momentum of the satellite is

$$H_s = I\dot{\theta} \tag{2.36}$$

where

$$\begin{aligned} I &= \text{inertia of satellite} \\ J &= \text{inertia of wheel} \\ \Omega &= \text{angular rate of wheel relative to the satellite} \\ \dot{\theta} &= \text{angular rate of satellite} \end{aligned}$$

Taking the time derivative of the sum of Eqs. 2.35 and 2.36 results in torque.

$$\frac{d}{dt}(H_w + H_s) = T \tag{2.37}$$

or

$$J\ddot{\theta} + J\dot{\Omega} + I\ddot{\theta} = T \tag{2.38}$$

where T is the external torque disturbance applied to the satellite.

The total torque the satellite experiences is the sum of external disturbances and the controlling torque generated by the momentum wheel. Taking the Laplace transform of Eq. 2.38 and adding in the controlling torque results in

$$Js^2\bar{\theta} + Js\bar{\Omega} + Is^2\bar{\theta} = \bar{T} + \bar{T}_c \tag{2.39}$$

where T_c is the controlling torque, the bar represents transformed quantities, and initial conditions are assumed to equal zero.

Eq. 2.39 is expressed in terms of the satellite only as

$$Is^2\bar{\theta} = \bar{T} + \bar{T}_c \tag{2.40}$$

and subtracting Eq. 2.40 from Eq. 2.39 generates the wheel's equation as follows:

$$Js^2\bar{\theta} + Js\bar{\Omega} = -\bar{T}_c \tag{2.41}$$

Eq. 2.40 is solved for $\bar{\theta}$ by dividing through by Is^2.

$$\bar{\theta} = \frac{\bar{T} + \bar{T}_c}{Is^2} \tag{2.42}$$

A simple control loop is illustrated in Figure 2.8.

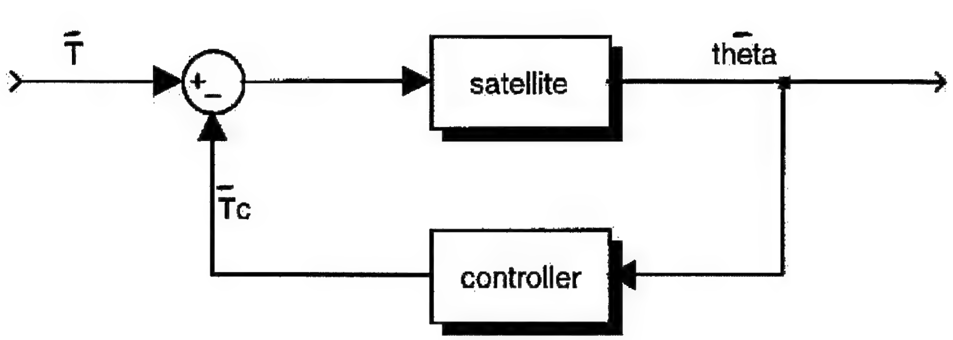


Figure 2.8 Momentum Wheel Control System.

where the satellite block represents Eq. 2.42 and the controller block is a PD controller with proportional (K) and derivative ($\frac{Ks}{\sigma}$) components. The equation for the PD controller becomes

$$K(\frac{s}{\sigma}+1) \tag{2.43}$$

or

$$K_R s + K_P \tag{2.44}$$

where K_R and K_P represent the rate and position gains respectively and σ is a condition for critical damping as follows:

$$\sigma = \frac{1}{2}\sqrt{\frac{K}{I}} \tag{2.45}$$

To calculate the gains of the controller, let

$$K = \frac{I}{\tau^2} \tag{2.46}$$

where τ is the time constant, or settling time, of the system.

2.6 Summary

The methods used to describe the rotational kinematics of a rigid body are direction cosines, Euler angles, and quaternions. Euler's rotational equations of motion provide the kinetic set of equations. The motion of an object is fully described when the kinematic and kinetic equations are combined. These equations also provide stability information of the object. A detailed explanation on how *SIMSAT* uses Euler angles and Euler's rotational equations to solve for inertial orientation is found in Section 4.3.6.

A stable environment must exist for satellites to meet mission pointing requirements. Spin, dual-spin, gravity gradient, and three-axis control are used to stabilize satellites. *SIMSAT* is a three-axis controlled satellite which uses momentum wheel control theory for stabilization. The system gains are calculated based on desired response times. Section 4.3.2 describes the block diagram representation of *SIMSAT*'s momentum wheel controller.

III. Experimental Model

3.1 Overview

SIMSAT is a laboratory-based satellite simulator. This simulator is a useful tool for instructors while teaching attitude control concepts. In addition, *SIMSAT* is capable of supporting several areas of research in attitude control and spacecraft stabilization. It is constructed out of several hardware and software components. Figure 3.1 illustrates where some of these components are located. The reader may use this figure to identify the location of each component while reading this chapter. The sections are divided up based on structural components, functional components, and the software used to execute *SIMSAT* experiments. A detailed description of each component is located within this chapter.

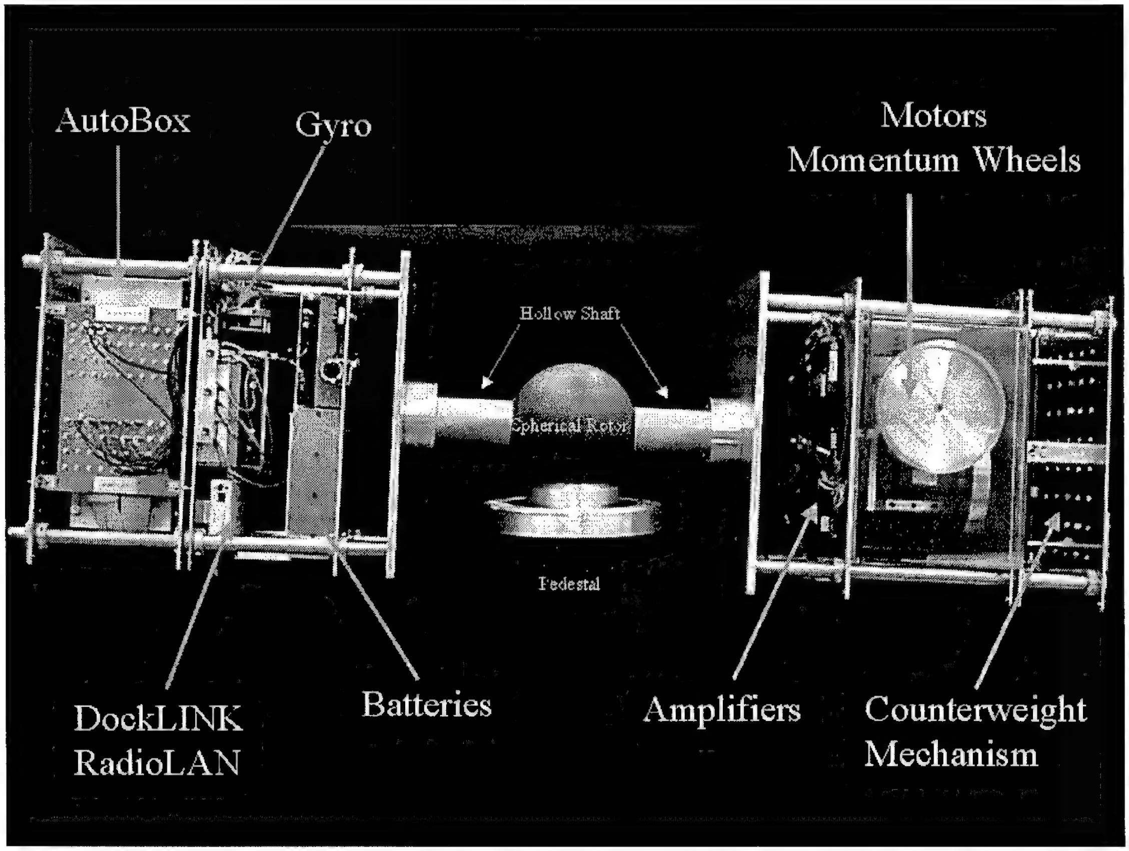


Figure 3.1 Components of *SIMSAT*

3.2 Structural Components

Figure 3.2 illustrates the basic *SIMSAT* structure. This structure supports individual components and acts as a skeleton for the entire system. The *SIMSAT* structure consists of the central sphere, hollow mounting shaft, and two box trusses which attach to each side of the mounting shaft. The central sphere and the mounting shaft are combined to form the air bearing assembly and has a mass of 19.32 kg (42.5 lbs). The box trusses are comprised of two baseplates, six mounting plates and eight mounting rods. These trusses house the various components on movable mounting plates. This allows *SIMSAT* components to be repositioned by the researcher to accommodate for emerging configurations and weight distribution. The movable mounting plates also provide easy access to various components. The total mass of the box trusses is 37.5 kg (82.5 lbs).

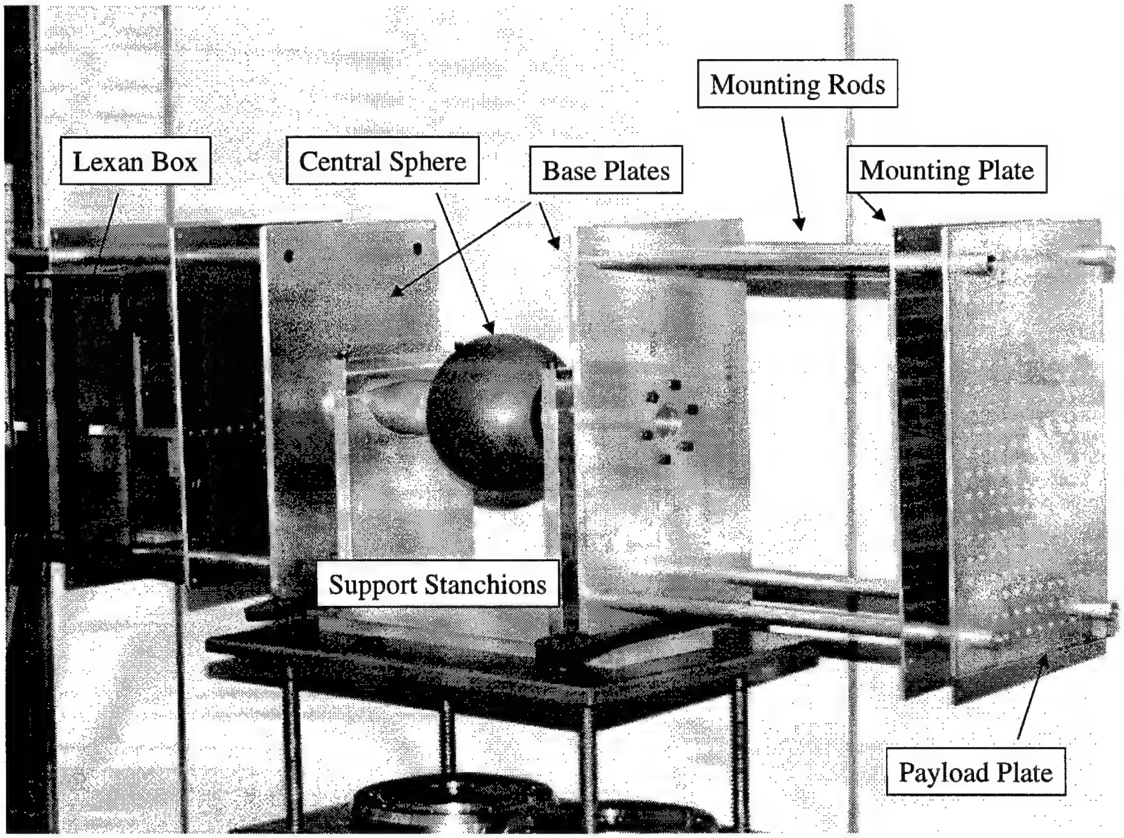


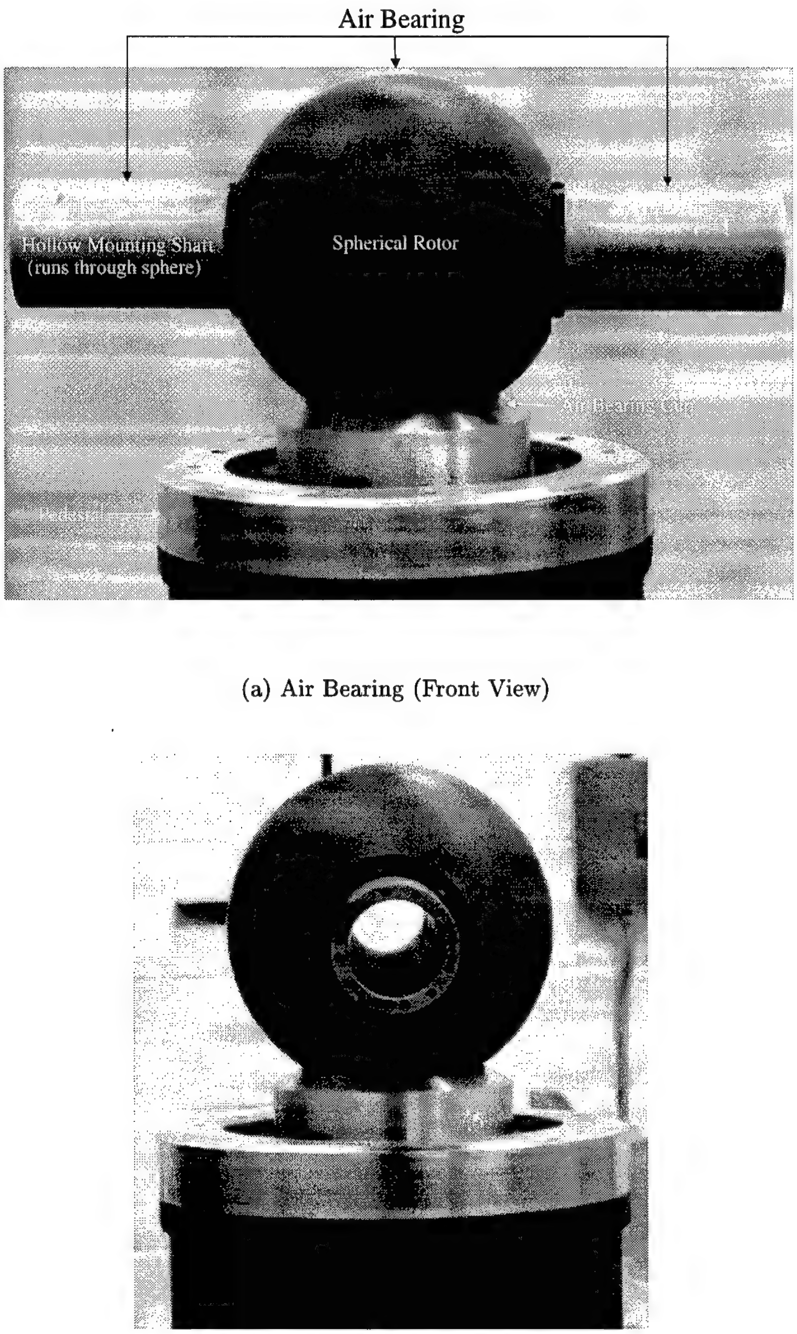
Figure 3.2 Partially-Assembled *SIMSAT* in the Laboratory

3.2.1 Air Bearing Assembly.

The air bearing assembly and its operating instructions are described in [29]. An air compressor supplies air at approximately 75 psi to the pedestal. This air is directed through six air jets positioned in the air bearing cup. The spherical rotor floats above the cup on a thin film of air measuring less than 0.0005 inches thick (see Figure 3.3). The air bearing is capable of 360° of yaw (rotation about the vertical axis), 360° of roll (rotation about the horizontal axis), and ±30° of pitch (tilt in the vertical plane). The air jets are capable of levitating at least 372.5 lbs. *SIMSAT* weighs approximately 310 lbs when fully constructed. A portable crane, shown in Figure 3.4, is used to lift *SIMSAT* and place it on top of the air bearing cup. Nylon web straps are wrapped around the hollow mounting shaft on both sides of the spherical rotor. Caution is required to prevent contact with the spherical rotor's teflon surface and the nylon straps. Small scratches on the spherical rotor's surface may apply spurious torques to *SIMSAT* while operational.

Two more precautionary steps help prevent damage to the spherical rotor's surface. It is important to clean the spherical rotor and the air bearing cup with denatured alcohol prior to placing the air bearing assembly on the pedestal. To prevent contamination from body oils, the researcher must never touch the rotor and cup with bare hands.

When the air bearing is not in use a metal plate covers the air jet cup to protect it from dust and scratches. A box made of Lexan sheet is placed around the spherical rotor to protect the teflon surface as well. The *SIMSAT* structure is placed on its support stanchions and the 'anti-tip' collars prevent the structure from toppling when heavy components on the truss are moved (see Figure 3.5).



(a) Air Bearing (Front View)

(b) Air Bearing (Side View)

Figure 3.3 Air Bearing

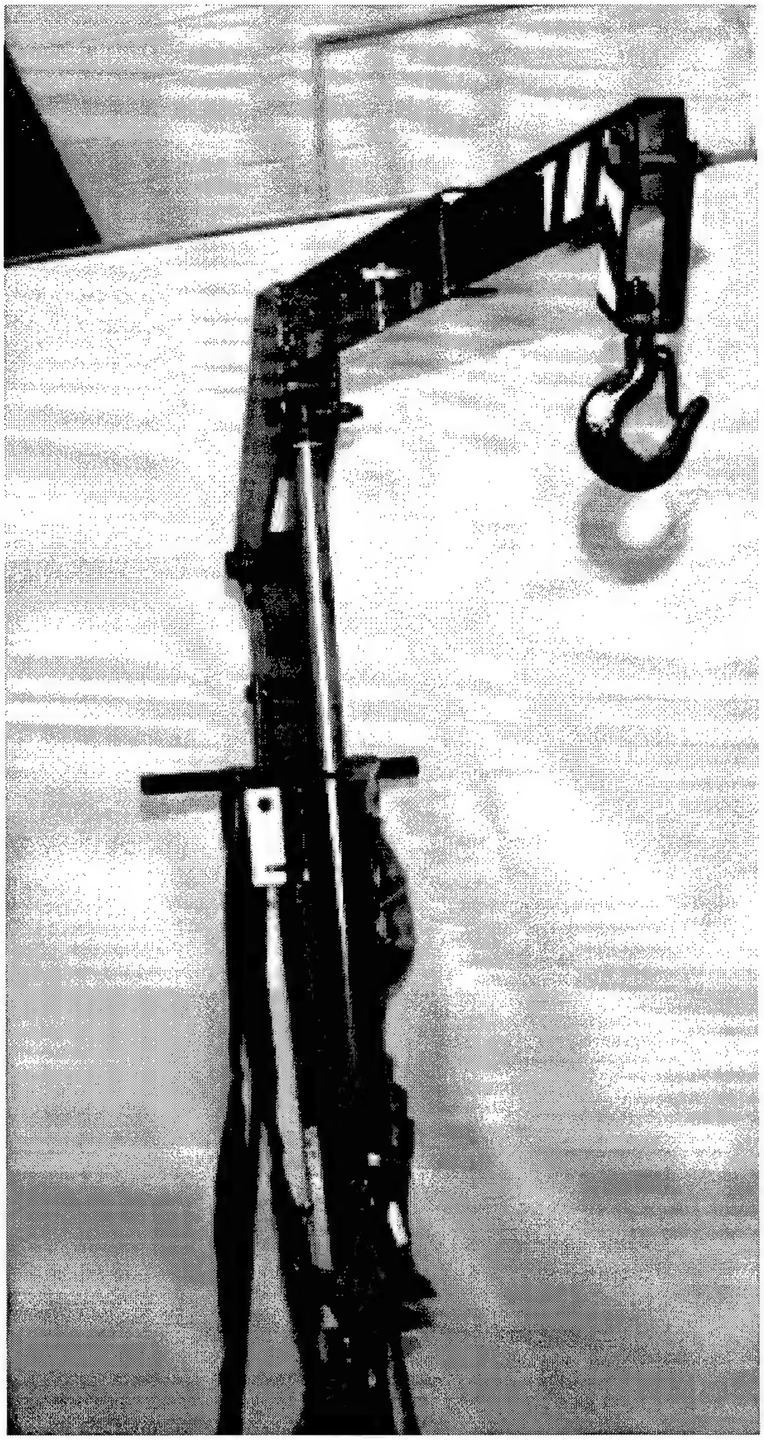


Figure 3.4 Portable Hydraulic Crane

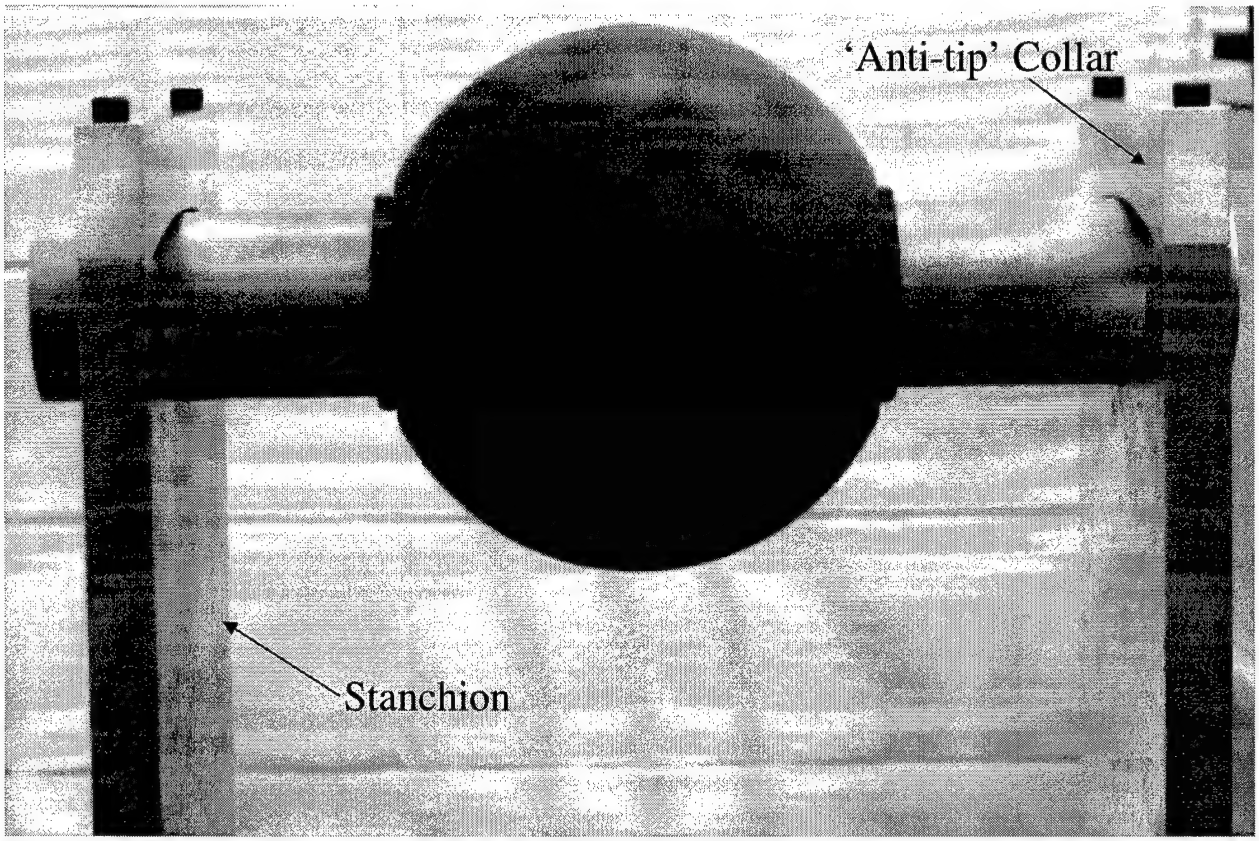
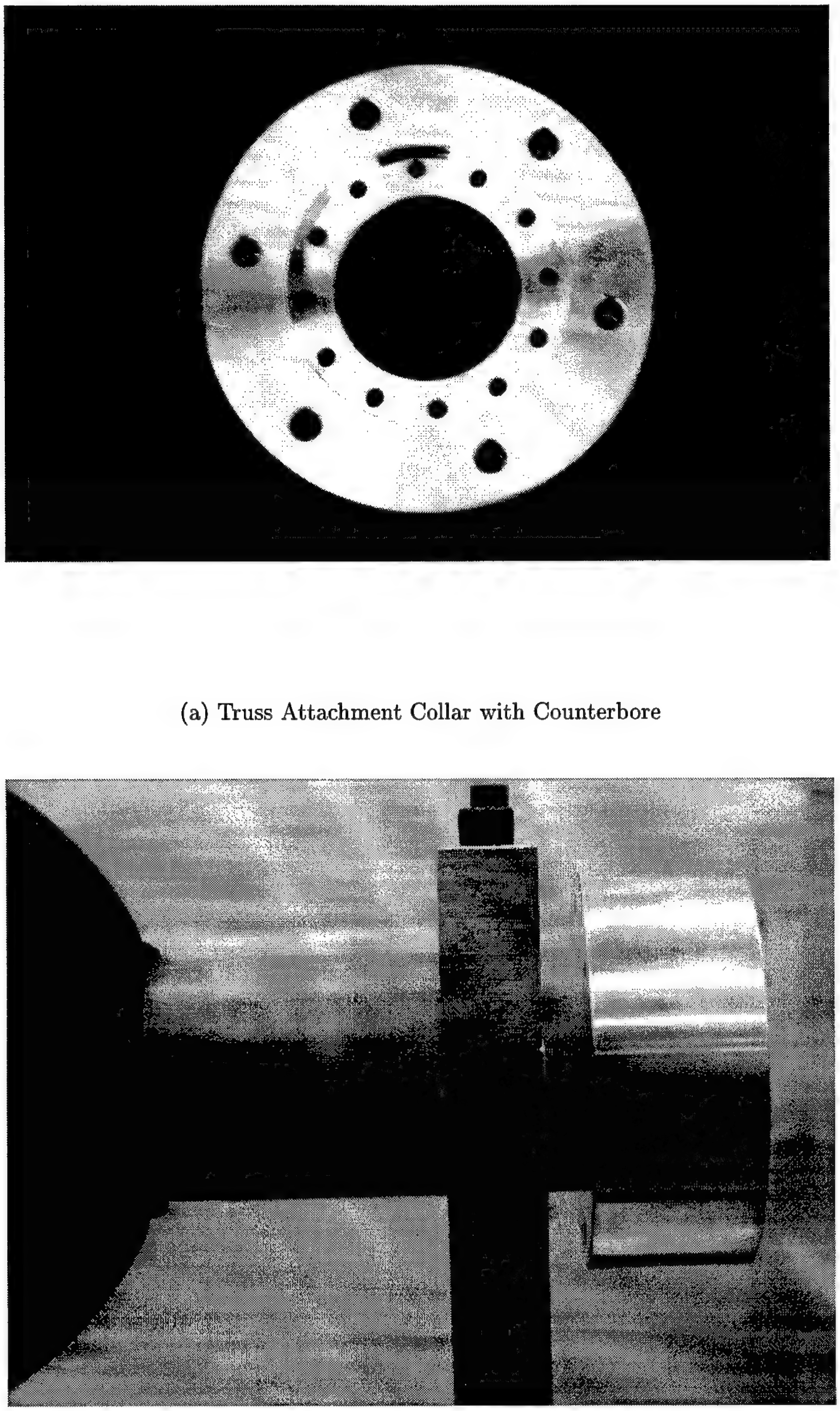


Figure 3.5 Air Bearing, Support Stanchions and 'Anti-tip' Collars

3.2.2 Truss Attachment Collars. An aluminum 7075-T7 collar is used to attach each box truss to the air bearing assembly. Each cylindrical collar has a 4.875" outer diameter and a 2" thickness. Amplifier data and power cables are routed through the hollow mounting shaft and the spherical rotor through a 2" diameter hole centered in the collar. Each collar also has a counterbore (3/16" deep and 3" diameter) designed to overlap the air bearing's mounting shaft. This counterbore overlap reduces the shear stress on the collar-to-mounting-shaft attachment screws (see Figure 3.6).



(a) Truss Attachment Collar with Counterbore

(b) Collar Attached to Mounting Shaft

Figure 3.6 Truss Attachment Collar

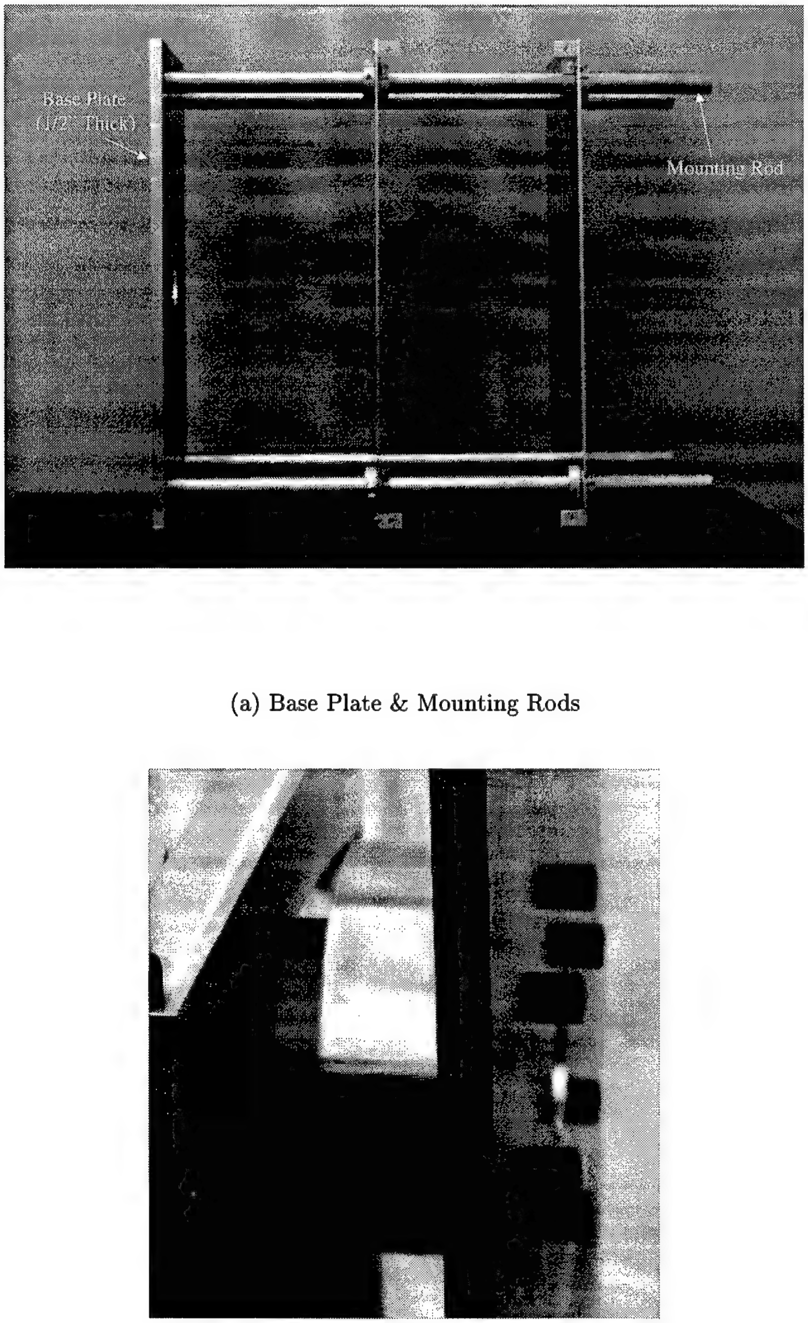
3.2.3 Base Plates and Mounting Rods.

The two 1/2" thick base plates allow for the attachment of each truss to its respective collar. The main truss support rods are 60 cm (23.6") long with an outer diameter of 1" and a wall thickness of 0.065". Each of the eight rods is constructed from stainless steel 316 tubing with one end plugged with a metal insert. The plugged end of each rod is placed through the base plate and mated with a connecting bolt to secure the rod in place. This arrangement allows for minimal protrusion on the inside of the base plate, thereby lessening interference with the overall *SIMSAT* pitch envelope (see Figure 3.7).

3.2.4 Mounting Plates.

Figure 3.8 shows a few of *SIMSAT*'s mounting plates. The various components of *SIMSAT* are attached to these plates. A standard template is used for all plates, which are available in a variety of thicknesses (1/2", 1/4", 3/16", 1/8", and 3/32"). Aircraft-grade aluminum is used in all instances: aluminum 7075-T6 for 1/8" and 3/16", aluminum 7075-T7 for 3/32" plates, and aluminum 2024 for 1/2" and 3/16" plates. Various plate thicknesses may be used on *SIMSAT* to reduce structural weight if plate buckling limits are not exceeded.

All plates are 53 cm (20.866") tall by 35 cm (13.78") wide. Four 1" diameter holes are located on the corners of each plate with centers offset 4.4 cm (1.732", vertically and horizontally) from the outer edgeline. These holes allow the mounting plates to slide onto the main truss support rods. Four 10-32 sized holes (centers located 0.5" in vertically and horizontally from edgeline) provide mounting points for L-bracket attachments. These L-brackets allow for diagonal cross-members to be attached between mounting plates (see Figure 3.9).



(a) Base Plate & Mounting Rods

(b) Base Plate Attached to Collar

Figure 3.7 *SIMSAT* Truss Base Plate & Mounting Rods

The mounting plates are fixed to their positions along the support rods through the use of metal clamp-on collars (with a 1" bore hole). Each mounting plate has a total of eight collars (one collar for each side of the four mounting holes) to prevent movement of the mounting plate along the support rod (see Figure 3.9). Researcher can adjust and secure the mounting plates to different points along the support rods to accommodate equipment changes and/or gross balance requirements. One-piece collars are used primarily for mounting plates not subject to frequent adjustment, while two-piece collars allow for easy take-on/take-off in situations where access is routinely required. Each collar is made from aluminum; a two piece collar weighs 0.04 kg and a one piece collar weighs 0.035 kg (weight includes the cap screws). The cap screws used to tighten the clamp-on collars are manufactured from alloy steel.

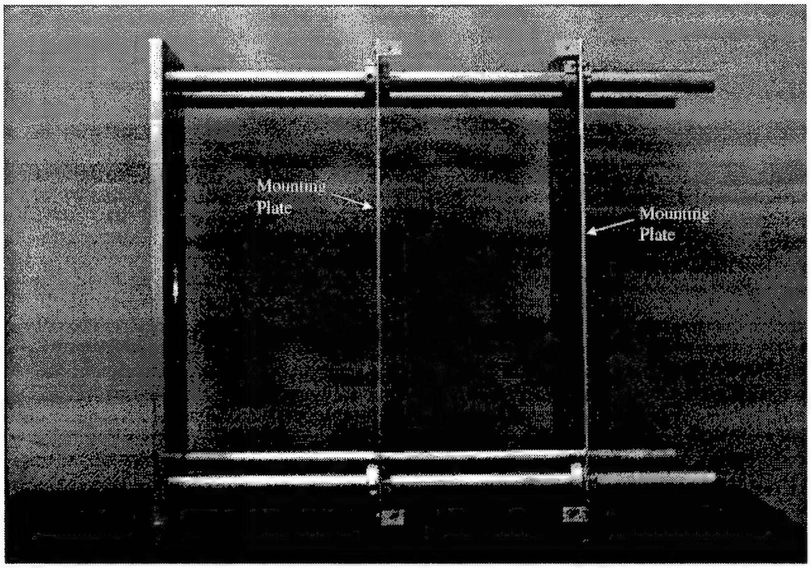


Figure 3.8 Mounting Plates on Truss

3.2.5 Payload Pegboard Plate. The payload plate is 1/4" thick with 212 threaded holes (5/16" diameter) spaced 1" apart (horizontally and vertically) between centers. These threaded holes are referred to collectively as a "pegboard" surface. The pegboard allows the user to attach experimental payloads or balancing weights to the

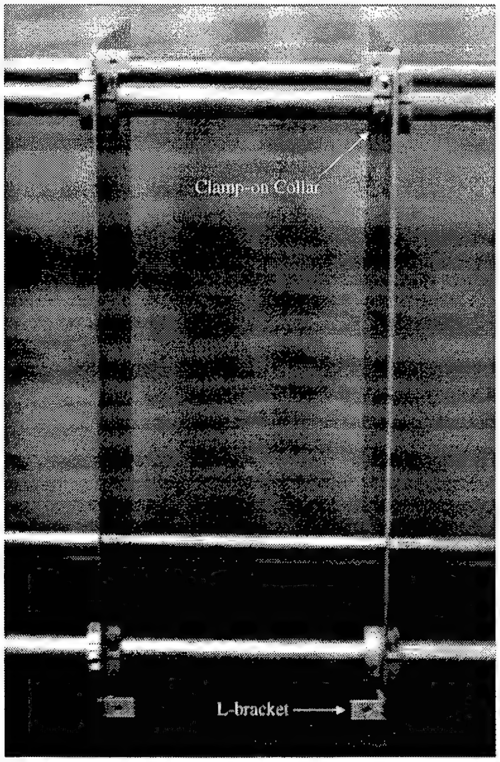


Figure 3.9 L-brackets and Clamp-on Collars

payload plate with a maximum amount of flexibility. Carbon steel blocks of nominal 5 kg and 1 kg masses allow for gross balancing of the *SIMSAT* structure. These blocks are attached to the pegboard using 5/16" bolts. The researcher attaches experiments directly to the pegboard with bolts or indirectly using mounting brackets attached to the experiment (see Figure 3.10).

3.2.6 Steel Counterweight Plates.

In addition to carbon steel blocks attached to the payload plate, gross *SIMSAT* balancing is accomplished with 53cm by 35cm steel counterweight plates. Four 2 kg counterweight plates are available and four 5 kg plates are available. These plates are cut from the standard plate template except they have lightening holes to provide the specified mass. The 2 kg plates are nominally 1/16" thick with a 6.72" diameter hole in the center. The 5 kg plates are nominally 3/16" thick with a 9.9" hole in the center. These plates give the user flexibility to place counterweight near the base plate (closer in towards the central sphere) to reduce inertia penalties.

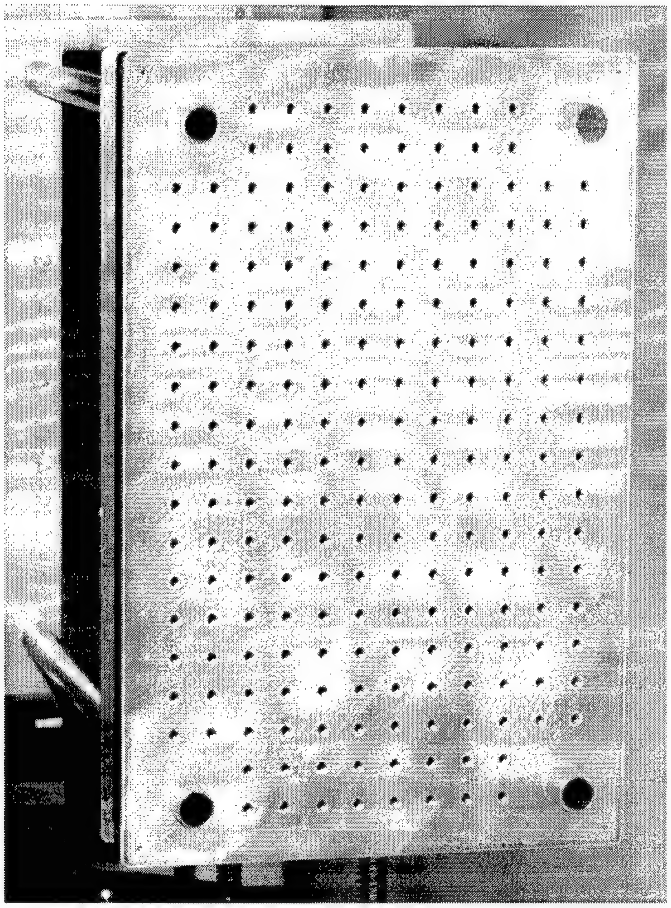


Figure 3.10 Payload Pegboard Plate

3.2.7 Fine-Tuning Counterweight Mechanism. Fine-tuning of *SIMSAT* static balance is accomplished with a counterweight mechanism (see Figure 3.11). The researcher can mount the counterweight mechanism on either side of *SIMSAT* depending on balancing needs. This mechanism relies upon orthogonal 1/4" stainless steel threaded rods, hollow cylindrical weights (each with a 1/4" hole), and small steel clamp-on collars (1/4" bore). The hollow cylindrical weights slide over the threaded rods and are held in place with the small clamp-on collars. Six 100 gram and five 500 gram hollow weights are available. Hand knobs on the ends of the threaded rods allow the researcher to make slight adjustments in weight position by turning clockwise or counterclockwise. Figure 3.12

illustrates the static balancing of *SIMSAT* without a payload. Gross balancing is accomplished by placing the momentum wheels and fine-tuning mechanism on one side of the *SIMSAT* truss and the remaining components are placed on the other side of the truss. Precision balancing is then accomplished by adding weight, as needed, to the fine-tuning mechanism's threaded rods and turning the hand knobs.

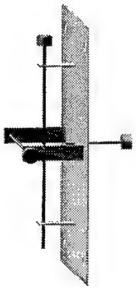


Figure 3.11 Fine-Tuning Counterweight Mechanism

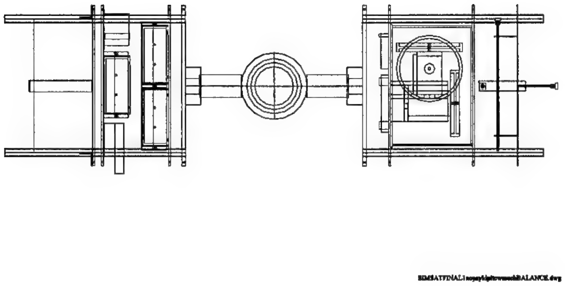


Figure 3.12 *SIMSAT* Balancing without Payload

3.2.8 Momentum Wheel Shelves and Lexan Box. The momentum wheel motors are attached to a mounting plate using a cantilevered support 'shelf' structure. Figure 3.13 shows the shelf assembly with one motor mounted.

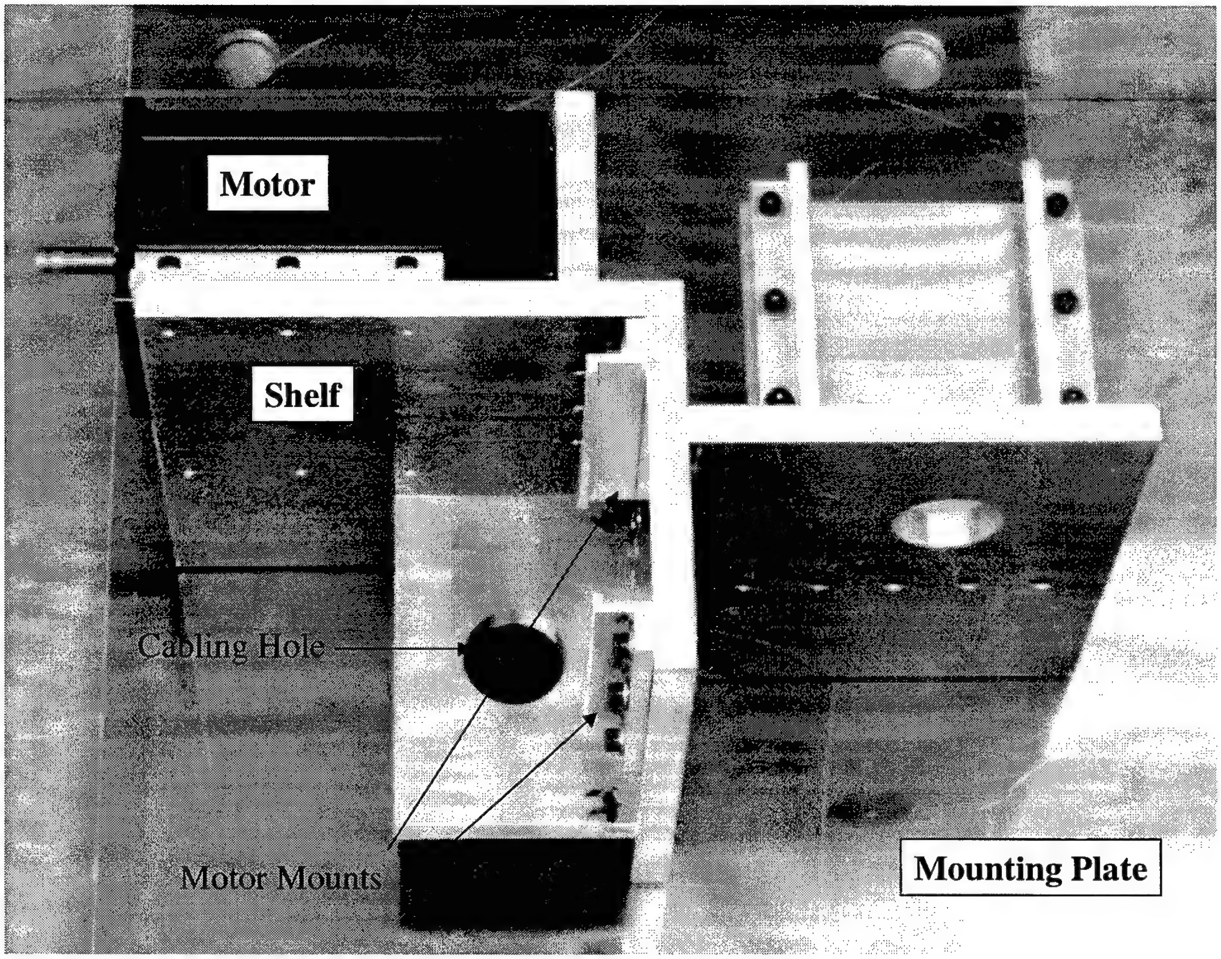


Figure 3.13 Momentum Wheel Motor Shelf Assembly

A safety housing encloses the momentum wheels on all sides. This prevents loose objects from being ejected by rotating parts. The housing consists of a six-sided Lexan box which extends outside of the truss bay in the z-direction (see Figure 3.14). Five sides of the box are attached together and affixed to a separate mounting plate. The sixth side of the box consists of a Lexan sheet mounted to the same plate as the momentum wheel motors. During *SIMSAT* assembly, the five-sided Lexan box 'slides' over the momentum wheels until a solid press fit is achieved. All Lexan sheets used to construct the box are 0.220" thick. Clearance between the momentum wheels and the interior of the Lexan box is approximately 0.5" to 1.0".

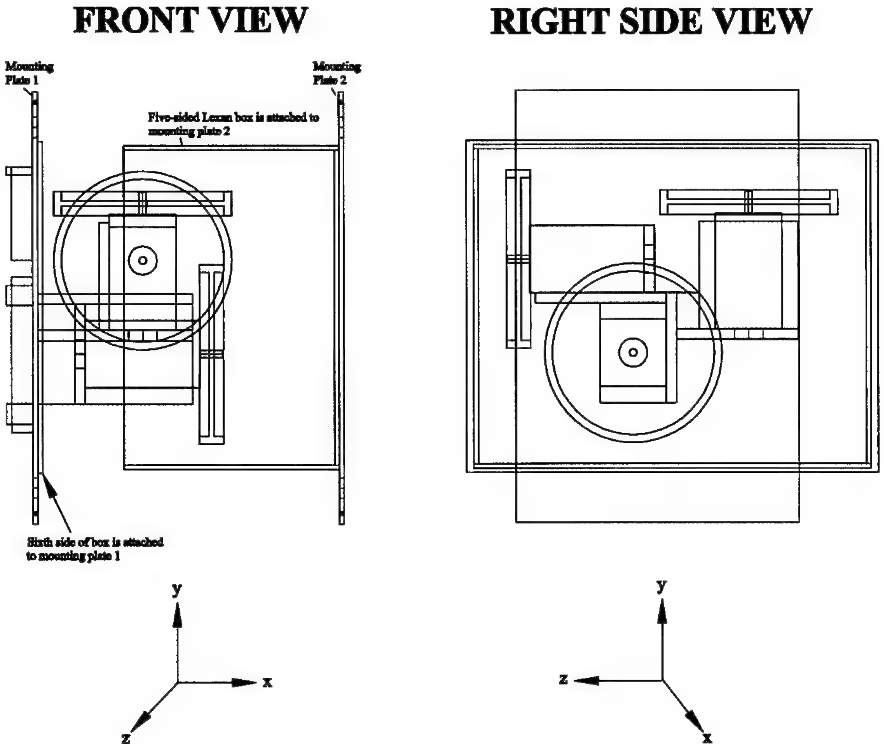

FRONT VIEW
RIGHT SIDE VIEW
Mounting Plate 1
Mounting Plate 2
Five-sided Lexan box is attached to mounting plate 2
Sixth side of box is attached to mounting plate 1
y
x
z
y
z
x

Figure 3.14 Lexan Box

3.3 Functional Components

3.3.1 Functional Overview. The functional components of *SIMSAT* provide the means for the vehicle to perform its primary functions. Within the system design, the three main functional subsystems include:

- **ADACS** - Attitude Determination and Control Subsystem
- **C&DH** - Command and Data Handling Subsystem
- **Power** - Power Subsystem

ADACS Within this subsystem, there are two main functional components–gyroscope and momentum wheels. The Humphrey CF-75 Series Axis Rate Gyro provides rate and acceleration data from which *SIMSAT*'s position can be derived. Momentum wheels provide the means to maintain a desired position or change the vehicle's orientation. The main components of the momentum wheels include three amplifiers, three motors, and three momentum wheels.

C&DH Within this subsystem, there are three main functional components-AutoBox, transmitter/receiver package, and signal interface plates. The AutoBox contains the computer processors and software codes necessary to run *SIMSAT*. The transmitter/receiver package allows *SIMSAT* to interact with the ground station. The signal interface plates provide the necessary input and output connections from the functional components to AutoBox.

Power Within this subsystem, there are four main components–power architecture, batteries, undervoltage alarm, and the power interface for the experimental payloads. The power architecture includes the wiring and equipment necessary to provide a 12 VDC, 24 VDC, and 36 VDC power supply from the batteries to the vehicle. The batteries are the voltage source. The undervoltage alarm informs experimenters when the batteries are running low. Finally, a special power interface allows the user to attach the experimental payloads to the existing power system.

3.3.2 Humphrey CF-75 Series Axis Rate Gyro. The gyro is used for attitude determination.

Vendor: Humphrey, Inc., 9212 Balboa Ave, San Diego, CA 92123

Model No: CF75-0201-1

Point of Contact: Mr. Bob Jones

Phone Number: (619) 565-6631

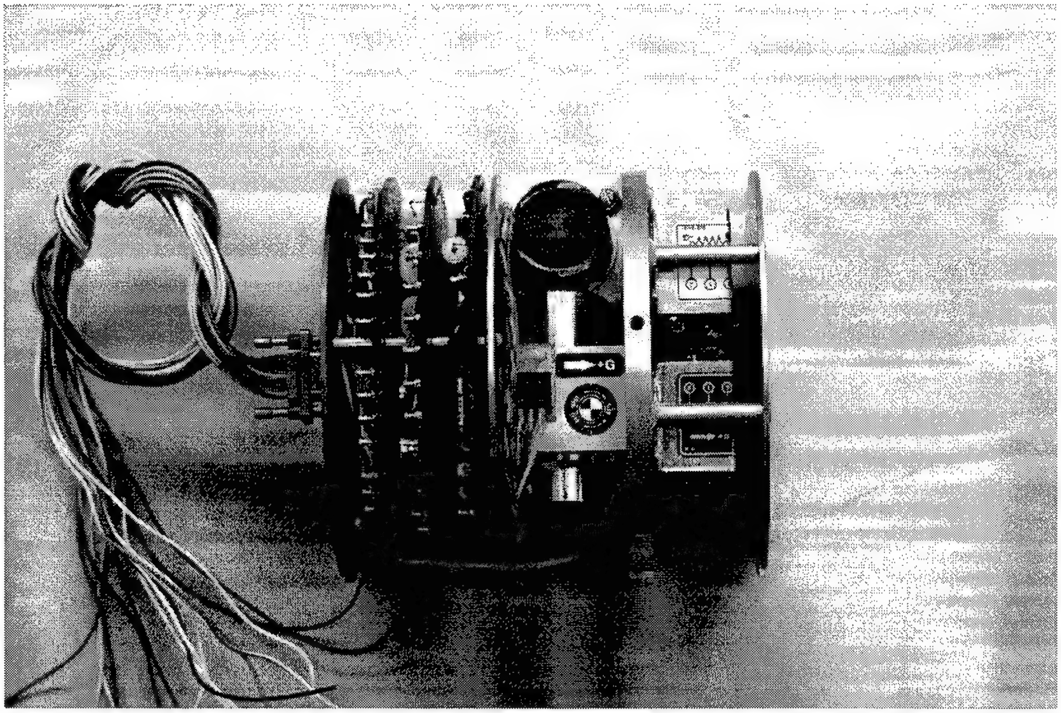


Figure 3.15 Humphrey CF75 Series Axis Rate Gyro

3.3.2.1 Characteristics. The Humphrey gyro has rate gyros for the roll, pitch, and yaw axes. Additionally, it has linear accelerometers which measure fore/aft acceleration, lateral acceleration, and vertical acceleration. The rates were chosen to account for the possible addition of thruster packages or other actuation systems which could significantly improve the slew capabilities of *SIMSAT*. Table 3.1 shows the ranges and accuracies for the roll, pitch, and yaw axes.

Table 3.1 Humphrey CF75 Characteristics

Parameter	Value
Operating Voltage	28±4V
Operating Current	0.58A
Weight	1.05 kg
Roll Rate Range	±120 deg/sec
Roll Accuracy (Half Range)	1.2 deg/sec
Roll Accuracy (Full Range)	4.8 deg/sec
Pitch/Yaw Rate Range	±40 deg/sec
Pitch/Yaw Accuracy (Half Range)	0.6 deg/sec
Pitch/Yaw Accuracy (Full Range)	2.4 deg/sec

Half Range is defined as operating within the first half of the operating range of the gyro. For example, in the roll axis, the gyro is capable of measuring slews from ±120 deg/sec. The half range is ±60 deg/sec. Based on historical data for this gyro series, the half range accuracy is rated as 0.5% of the full scale to half range. In other words, the full scale of the roll axis is 240 deg/sec. The accuracy, then, is 0.5% of 240 deg/sec, or ±1.2 deg/sec.

Half to Full Range is defined as operating within the second half of the operating range of the gyro. Once again, using the roll axis, the gyro is capable of measuring slews from ±120 deg/sec. The half to full range is rated from ±60 deg/sec to 120 deg/sec. Based on historical data for this gyro series, the performance of the gyro changes. The accuracy is rated at 2% of the full scale to half to full range. This corresponds to an accuracy of ±4.8 deg/sec. Mr. Bob Jones said the accuracies are padded estimations. The actual accuracies may be much better than what is listed.

Additionally, if it is desired to adjust the roll, pitch, and yaw sensing rates, Humphrey Inc. can perform adjustments back at their factory. However, if the operational rate ranges exceed ±200 deg/sec in any axis, the factory must replace the hardware for that axis. Table 3.2 shows the expected accuracies for the different range rates.

Table 3.2 Sensing Ranges and Accuracies

Rate (deg/sec)	HR Accuracy (deg/sec)	Full Range Accuracy (deg/sec)
20	0.2	0.8
30	0.3	1.2
40	0.4	1.6
50	0.5	2.0
60	0.6	2.4
70	0.7	2.8
80	0.8	3.2
90	0.9	3.6
100	1.0	4.0
110	1.1	4.4
120	1.2	4.8
130	1.3	5.2
140	1.4	5.6
150	1.5	6.0
160	1.6	6.4
170	1.7	6.8
180	1.8	7.2
190	1.9	7.6
200	2.0	8.0

3.3.2.2 Mounting Considerations. Using the baseline design, the gyro is attached on the same mounting plate holding a single battery and the transmitter/receiver package (please refer to Figure 3.16).

3.3.2.3 Power Requirements/Interface. As stated in Table 3.1, the operational voltage requirement is 28±4 VDC and the operating current is 0.58 Amps. At startup, there is an initial draw of 50 Watts. However, following initialization of the gyro, the power consumption is only 25 Watts. The wiring schematics for the gyro is provided in Figure 3.18. Pins 3, 14, and 1 are used for input wiring. Each battery supplies approximately 13 VDC of power when fully charged. The power cable for the gyro is connected to the 24 VDC section of the power bus bar. Mr. Bob Jones ensures the gyro will function normally down to voltages of 22 VDC.

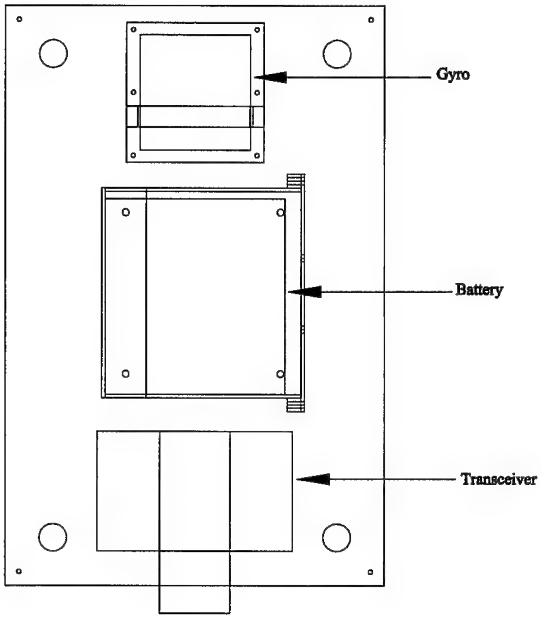
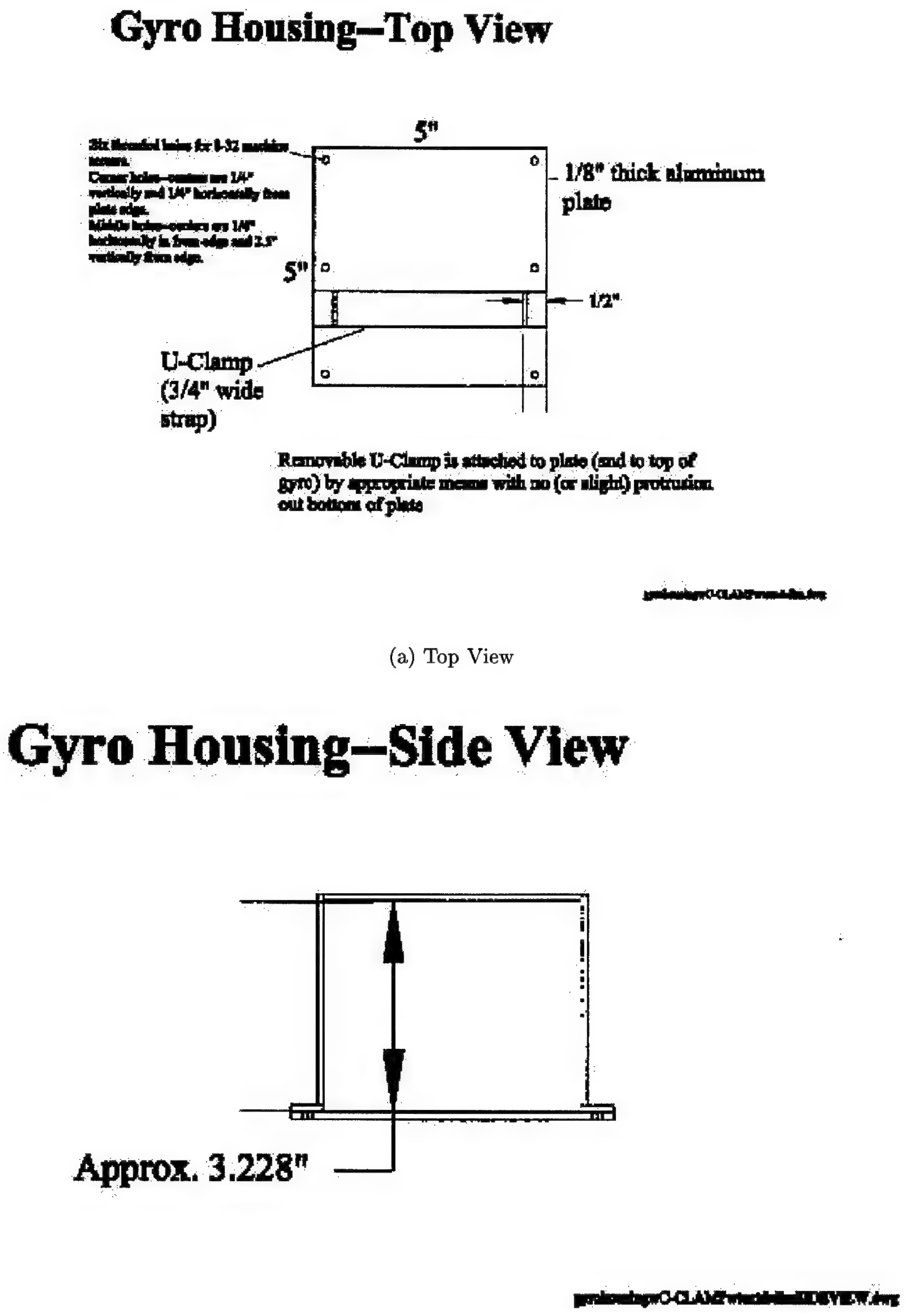


Figure 3.16 Plate Position for the Gyro

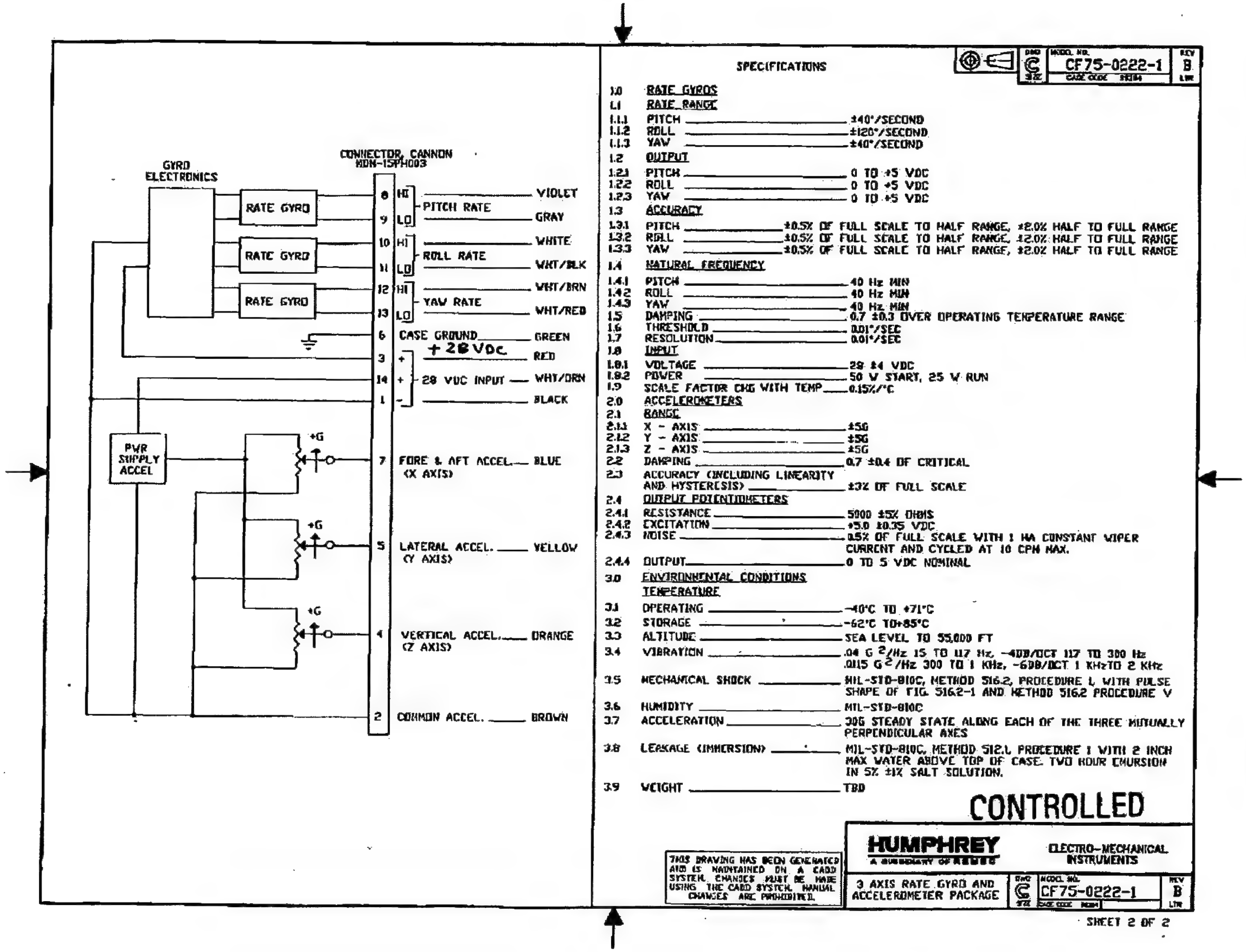

Gyro Housing–Top View
5"
Six threaded holes for 8-32 machine screws.
Corner holes–centers are 1/4" vertically and 1/4" horizontally from plate edge.
Middle holes–centers are 1/4" horizontally in from edge and 2.5" vertically from edge.
1/8" thick aluminum plate
5"
1/2"
U-Clamp (3/4" wide strap)
Removable U-Clamp is attached to plate (and to top of gyro) by appropriate means with no (or slight) protrusion out bottom of plate
Gyro Housing–Side View
Approx. 3.228"

(a) Top View

(b) Side View

Figure 3.17 Gyro Housing

FEB. 8.1999 3:37PM HUMPHREY INC NO.923 P.3



SPECIFICATIONS

MODEL NO. CF75-0222-1 REV B

1.0	RATE GYROS	
1.1	RATE RANGE	
1.1.1	PITCH	±40°/SECOND
1.1.2	ROLL	±120°/SECOND
1.1.3	YAW	±40°/SECOND
1.2	OUTPUT	
1.2.1	PITCH	0 TO +5 VDC
1.2.2	ROLL	0 TO +5 VDC
1.2.3	YAW	0 TO +5 VDC
1.3	ACCURACY	
1.3.1	PITCH	±0.5% OF FULL SCALE TO HALF RANGE, ±2.0% HALF TO FULL RANGE
1.3.2	ROLL	±0.5% OF FULL SCALE TO HALF RANGE, ±2.0% HALF TO FULL RANGE
1.3.3	YAW	±0.5% OF FULL SCALE TO HALF RANGE, ±2.0% HALF TO FULL RANGE
1.4	NATURAL FREQUENCY	
1.4.1	PITCH	40 Hz MIN
1.4.2	ROLL	40 Hz MIN
1.4.3	YAW	40 Hz MIN
1.5	DAMPING	0.7 ±0.3 OVER OPERATING TEMPERATURE RANGE
1.6	THRESHOLD	0.01°/SEC
1.7	RESOLUTION	0.01°/SEC
1.8	INPUT	
1.8.1	VOLTAGE	28 ±4 VDC
1.8.2	POWER	50 W START, 25 W RUN
1.9	SCALE FACTOR CHG WITH TEMP	0.15%/°C
2.0	ACCELEROMETERS	
2.1	RANGE	
2.1.1	X - AXIS	±5G
2.1.2	Y - AXIS	±5G
2.1.3	Z - AXIS	±5G
2.2	DAMPING	0.7 ±0.4 OF CRITICAL
2.3	ACCURACY (INCLUDING LINEARITY AND HYSTERESIS)	±3% OF FULL SCALE
2.4	OUTPUT POTENTIOMETERS	
2.4.1	RESISTANCE	5000 ±5% OHMS
2.4.2	EXCITATION	+5.0 ±0.35 VDC
2.4.3	NOISE	0.5% OF FULL SCALE WITH 1 MA CONSTANT WIPER CURRENT AND CYCLED AT 10 CPM MAX.
2.4.4	OUTPUT	0 TO 5 VDC NOMINAL
3.0	ENVIRONMENTAL CONDITIONS	
	TEMPERATURE	
3.1	OPERATING	-40°C TO +71°C
3.2	STORAGE	-62°C TO+85°C
3.3	ALTITUDE	SEA LEVEL TO 55,000 FT
3.4	VIBRATION	.04 G ^{2}/Hz 15 TO 117 Hz, -4DB/OCT 117 TO 300 Hz .0115 G ^{2}/Hz 300 TO 1 KHz, -6DB/OCT 1 KHzTO 2 KHz
3.5	MECHANICAL SHOCK	MIL-STD-810C, METHOD 516.2, PROCEDURE I, WITH PULSE SHAPE OF FIG. 516.2-1 AND METHOD 516.2 PROCEDURE V
3.6	HUMIDITY	MIL-STD-810C
3.7	ACCELERATION	30G STEADY STATE ALONG EACH OF THE THREE MUTUALLY PERPENDICULAR AXES
3.8	LEAKAGE (IMMERSION)	MIL-STD-810C, METHOD 512.1, PROCEDURE I WITH 2 INCH MAX WATER ABOVE TOP OF CASE. TWO HOUR EMERSION IN 5% ±1% SALT SOLUTION.
3.9	WEIGHT	TBD

CONTROLLED

THIS DRAWING HAS BEEN GENERATED AND IS MAINTAINED ON A CADD SYSTEM. CHANGES MUST BE MADE USING THE CADD SYSTEM. MANUAL CHANGES ARE PROHIBITED.

HUMPHREY
A SUBSIDIARY OF [illegible]

ELECTRO-MECHANICAL INSTRUMENTS

3 AXIS RATE GYRO AND ACCELEROMETER PACKAGE

MODEL NO. CF75-0222-1 REV B

SHEET 2 OF 2

Figure 3.18 Gyro Specifications

3.3.3 Advanced Motion Controls BE40A8 Servo Amplifier.

The amplifier controls the input to a given motor and measures the wheel speed via a tachometer.

Vendor: Servo Systems Co., 115 Main Road P.O. Box 97, Montville NJ 07045-0097

Manufacturer: Advanced Motion Controls 3629 Vista Mercado Camarillo, CA 93012

Model No: BE40A8

Vendor Point of Contact: Mr. James Werba

Vendor Phone Number: (800) 922-1103

Manufacturer Phone Number: (805) 389-1935

Manufacturer Web Site: www.a-m-c.com

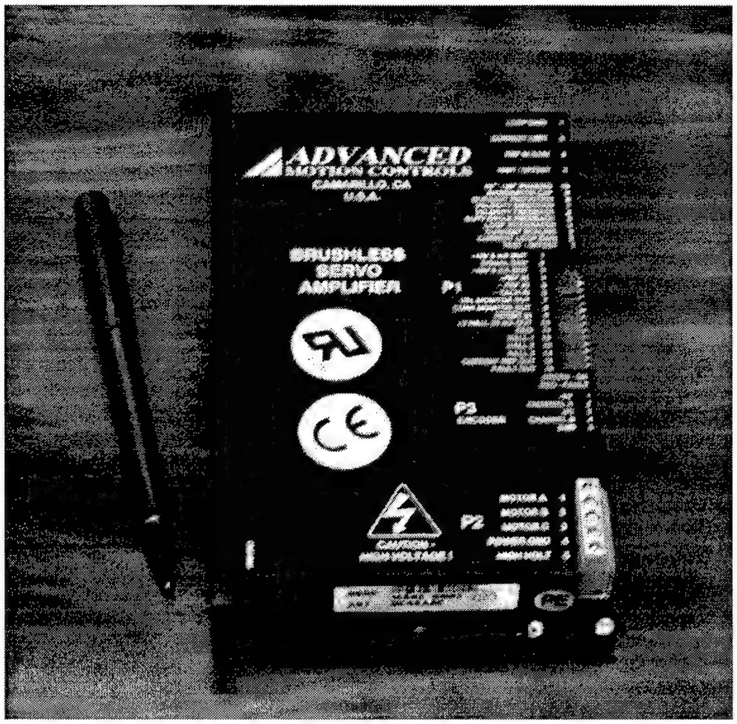


Figure 3.19 Advanced Motion Controls BE40A8 Servo Amplifier

3.3.3.1 ***Characteristics.*** Three Advanced Motion Controls BE40A8 Servo Amplifiers control the amount of torque provided by the BL-3450 motors based upon input signals from the gyroscope. The general characteristics of the amplifiers are listed in Table 3.3. While each amplifier is rated to +80 VDC, the BL-3450 motor is only rated up to +48 VDC.

Table 3.3 Advanced Motion Control Amplifier Characteristics

Parameter	Value
DC Supply Voltage	20-80V
Peak Current	±40A
Max. Continuous Current	±20A
Switching Frequency	22±15 KHz%
Bandwidth	2.5 KHz

3.3.3.2 ***Mounting Considerations.*** The amplifiers are attached to the momentum wheel plate on the opposite side of the momentum wheel shelves (please see Figure 3.20). They are oriented in such a fashion as to place the centroid of the three amplifiers on the b1 axis.

3.3.3.3 ***Power Requirements/Interface.*** The amplifiers are connected to the +36 VDC power supply. Although each amplifier is capable of using a +20 VDC to +80 VDC voltage source, the BL-3450 motor's operational range is between +20 VDC and +48 VDC. Therefore, the maximum voltage used by the amplifier is +48 VDC, which could be achievable if additional batteries are later added to the current design.

Using Table 3.3, the amplifier's peak current and maximum continuous current are ±40 A and ±20 A, respectively. The peak current is defined within a 2 second time interval. Both current ratings are internally limited due to components within the amplifier. Each amplifier has a power cable running through the hollow mounting shaft and connects to the 36 VDC section of the power bus bar.

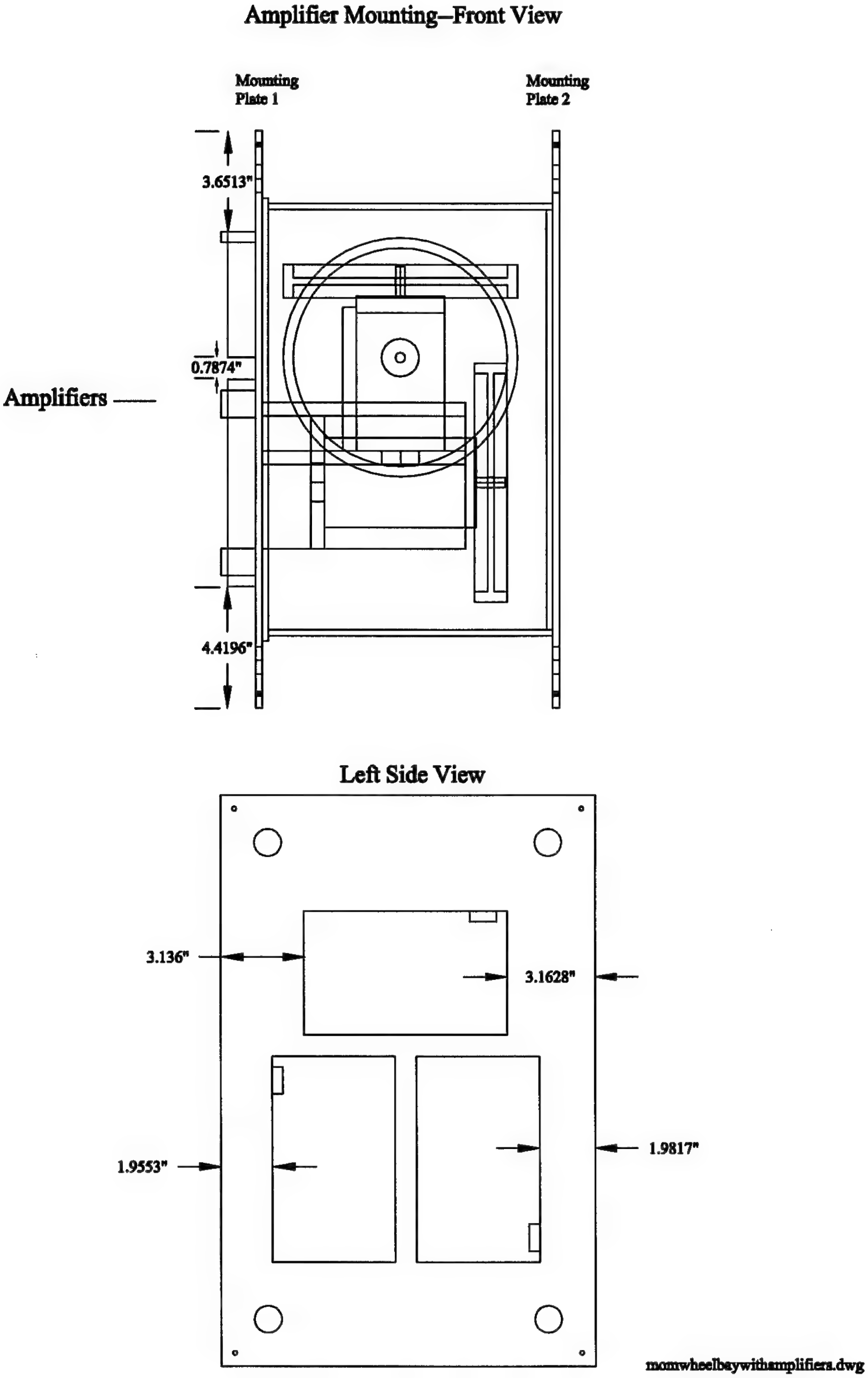
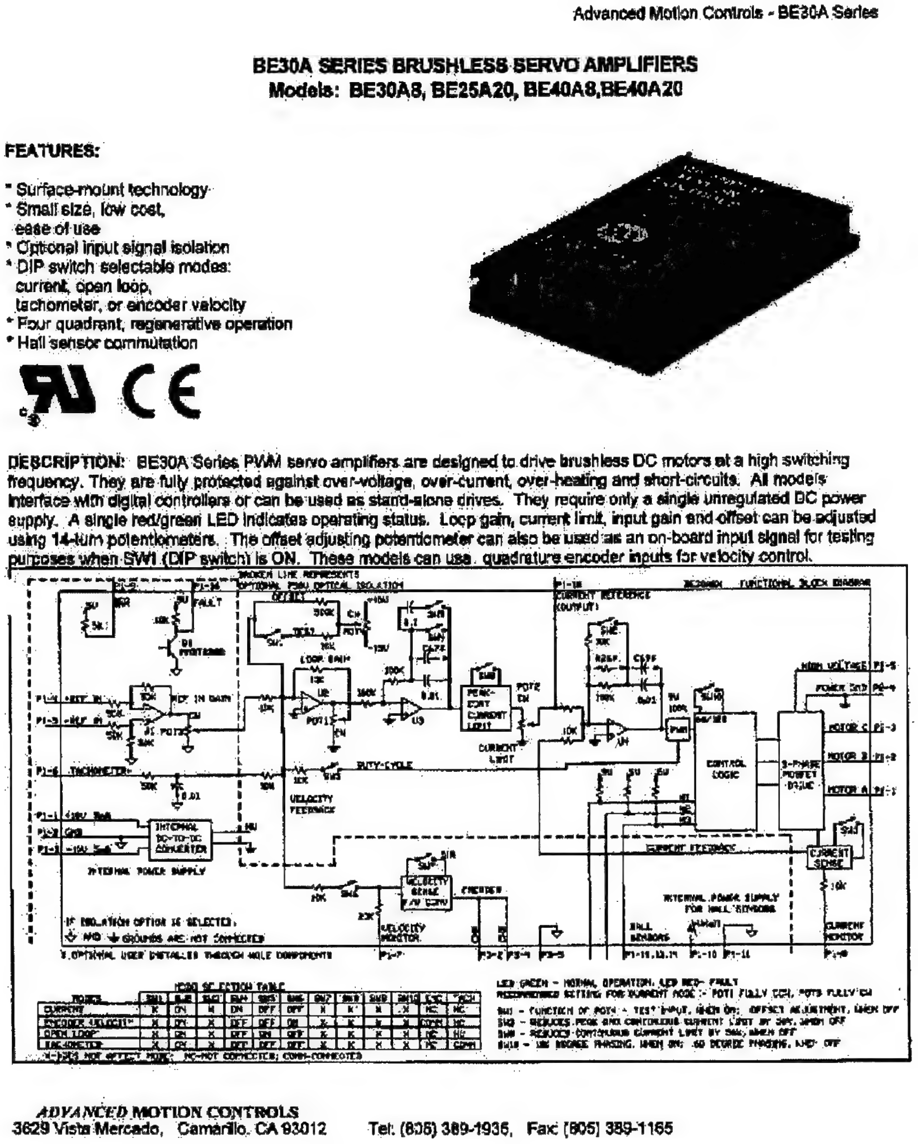


Figure 3.20 Amplifier Arrangement

3.3.3.4 Signal Requirements/Interface. The wiring schematic for each amplifier is shown in Figure 3.21. Table 3.4 shows the operational switch placements and Tables 3.5 and 3.6 describe the signal cable connection between each amplifier and its respective motor.



Advanced Motion Controls - BE30A Series

BE30A SERIES BRUSHLESS SERVO AMPLIFIERS
Models: BE30A8, BE25A20, BE40A8,BE40A20

FEATURES:

- Surface-mount technology
- Small size, low cost, ease of use
- Optional input signal isolation
- DIP switch selectable modes: current, open loop, tachometer, or encoder velocity
- Four quadrant, regenerative operation
- Hall sensor commutation

DESCRIPTION: BE30A Series PWM servo amplifiers are designed to drive brushless DC motors at a high switching frequency. They are fully protected against over-voltage, over-current, over-heating and short-circuits. All models interface with digital controllers or can be used as stand-alone drives. They require only a single unregulated DC power supply. A single red/green LED indicates operating status. Loop gain, current limit, input gain and offset can be adjusted using 14-turn potentiometers. The offset adjusting potentiometer can also be used as an on-board input signal for testing purposes when SW1 (DIP switch) is ON. These models can use quadrature encoder inputs for velocity control.

MODE SELECTION TABLE

MODES	SW1	SW2	SW3	SW4	SW5	SW6	SW7	SW8	SW9	SW10	ENC	TACH
CURRENT	X	ON	X	ON	OFF	OFF	X	X	X	X	NC	NC
ENCODER VELOCITY	X	ON	X	OFF	OFF	ON	X	X	X	X	COMM	NC
OPEN LOOP	X	ON	X	OFF	ON	OFF	X	X	X	X	NC	NC
TACHOMETER	X	ON	X	OFF	OFF	OFF	X	X	X	X	NC	COMM

X-DOES NOT AFFECT MODE; NC-NOT CONNECTED; COMM-CONNECTED

LED GREEN - NORMAL OPERATION, LED RED- FAULT
RECOMMENDED SETTING FOR CURRENT MODE :- POT1 FULLY CCW, POT3 FULLY CW

SW1 - FUNCTION OF POT4 - TEST INPUT, WHEN ON; OFFSET ADJUSTMENT, WHEN OFF
SW2 - REDUCES PEAK AND CONTINUOUS CURRENT LIMIT BY 50%, WHEN OFF
SW4 - REDUCES CONTINUOUS CURRENT LIMIT BY 50%, WHEN OFF
SW10 - 120 DEGREE PHASING, WHEN ON; 60 DEGREE PHASING, WHEN OFF

ADVANCED MOTION CONTROLS
3629 Vista Mercado, Camarillo, CA 93012 Tel: (805) 389-1936, Fax: (805) 389-1165

Figure 3.21 Wiring Schematic for the Advanced Motion Controls BE40A8 Servo Amplifier

Table 3.4 Advanced Motion Control Amplifier Switch Placement

Switch	Position
1	OFF
2	ON
3	ON
4	ON
5	OFF
6	ON
7	OFF
8	ON
9	OFF
10	ON

Table 3.5 Advanced Motion Control Amplifier Wiring Placement (amplifiers 1 and 2)

Location	Destination	Wire Color
P2 #1	Motor pin 1	Brown
P2 #2	Motor pin 2	Orange
P2 #3	Motor pin 3	Green
P2 #4	Power Bus Bar Ground	Black
P2 #5	Power Bus Bar Voltage	Red
P3 #1	Motor pin 4	Red
P3 #2	Motor pin 8	Yellow
P3 #4	Motor pin 10	Blue
P3 #5	Motor pin 5	Black
P1 #2	Input from Autobox ground	Copper
P1 #4	Input from Autobox	Copper
P1 #7	Input to Autobox	Copper
P1 #8	Input to Autobox	Copper
P1 #11	Input to Autobox grounds	Metal
P1 #12	Motor pin 12	White/Brown
P1 #13	Motor pin 13	White/Orange
P1 #14	Motor pin 14	White/Green

Table 3.6 Advanced Motion Control Amplifier Wiring Placement (amplifier 3)

Location	Destination	Wire Color
P2 #1	Motor pin 1	Orange
P2 #2	Motor pin 2	Brown
P2 #3	Motor pin 3	Green
P2 #4	Power Bus Bar Ground	Black
P2 #5	Power Bus Bar Voltage	Red
P3 #1	Motor pin 4	Blue/Black
P3 #2	Motor pin 8	Green/White
P3 #4	Motor pin 10	White/Black
P3 #5	Motor pin 5	Green/Black
P1 #2	Input from Autobox ground	Metal
P1 #4	Input from Autobox	Copper
P1 #7	Input to Autobox	Copper
P1 #8	Input to Autobox	Copper
P1 #11	Input to Autobox grounds	Metal
P1 #12	Motor pin 12	Orange (thin)
P1 #13	Motor pin 13	Blue/White
P1 #14	Motor pin 14	White

3.3.4 Animatics BL-3450 Brushless DC Servo Motor.

The motor provides the torque to move a momentum wheel.

Vendor: Servo Systems Co., 115 Main Road P.O. Box 97, Montville NJ 07045-0097

Manufacturer: Animatics Corporation, 3050 Tasman Drive, Santa Clara CA 95054-1116

Model No: BL-3450

Vendor Point of Contact: Mr. James Werba

Vendor Phone Number: (800) 922-1103

Manufacturer Point of Contact: Mr. Edmund Kong

Manufacturer Phone Number: (408) 748-8721

Manufacturer Web Site: www.animatics.com

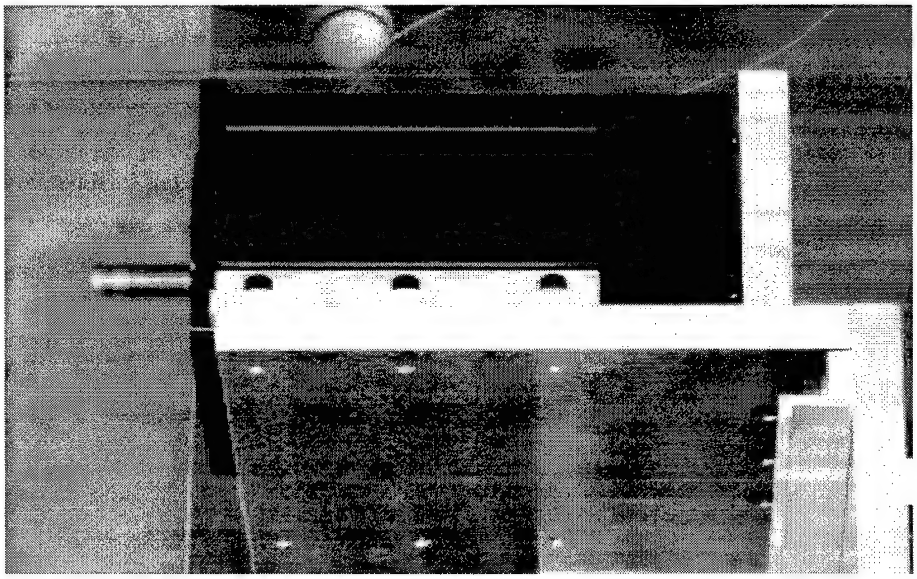


Figure 3.22 BL-3450 Brushless DC Servo Motor

3.3.4.1 Characteristics. AFIT purchased three BL-3450 Brushless DC Servo Motors, an integral part of the momentum wheel package, to provide torque to the momentum wheels. Each motor has the following characteristics listed in Table 3.7. The key characteristics are the torque values and the maximum no load speed. It should be noted these values were obtained at the factory using a 48 VDC source. Therefore, the torque values will be lower on *SIMSAT* while using a 36 VDC source. Due to the fact the momentum wheels are attached to the motor shafts, the actual speed is significantly reduced. After bench tests in the lab, the maximum wheel speed was determined to be approximately 1900 RPM.

Mr. Jay Anderson created a torque stand to measure the amount of torque provided by the momentum wheel as a function of voltage. Using this information, in concert with the wheel speed vs. voltage information, it is possible to develop torque vs. wheel speed, torque vs. input voltage, and wheel speed vs. input voltage curves from which we can derive motor models for the Simulink code.

Table 3.7 BL-3450 Motor Characteristics

Parameter	Value
Peak Torque	750 oz-in
Continuous Torque	250 oz-in
Voltage Constant	13.7V/kRPM
No Load Speed	3398 RPM
Max Load Speed	2012 RPM
Torque Constant	18.5 oz-in/Amp
Rotor Inertia	0.025 oz-in-sec^2
Weight	3.27 kg
Number of Poles	4
Number of Slots	24
Length	6.088 in

3.3.4.2 ***Mounting Considerations.*** As previously discussed in the hardware section, there are momentum wheel shelves used to brace the motors in place. This is illustrated in Figure 3.23.

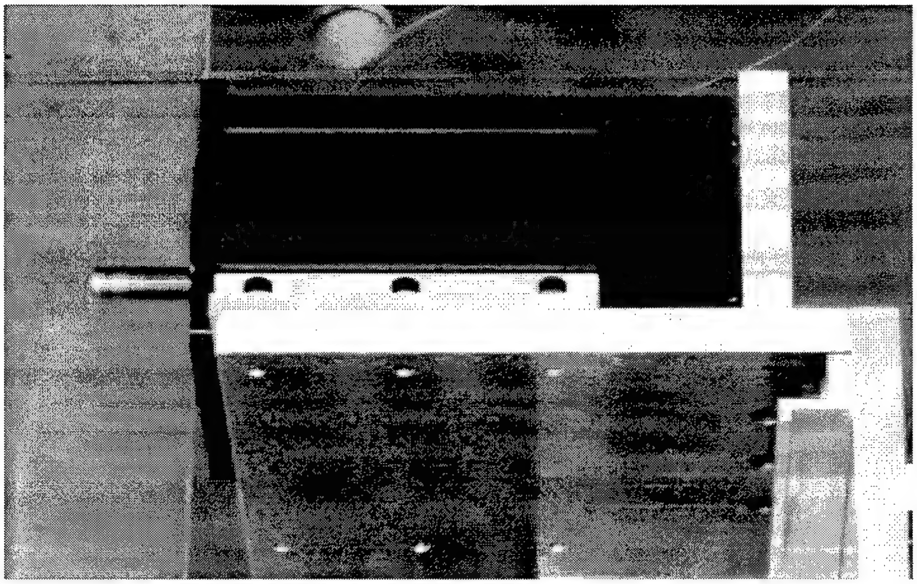


Figure 3.23 Motor Mounted to a Momentum Wheel Shelf

3.3.4.3 ***Power Requirements/Interface.*** The motor's operating voltage ranges from +20 VDC to +48 VDC. For *SIMSAT*, each motor is indirectly powered by a 36 VDC source through the Advanced Motion Controls BE408AU amplifier. Therefore, there is no need for a direct power cable running from the voltage source to the motor.

3.3.4.4 ***Signal Requirements/Interface.*** On the back of the motor, there is a female connector similar to the one shown in Figure 3.24. Table 3.8 lists the pin assignments. Mr. Bob Bacon successfully integrated the motor to the amplifier using these pin assignments in conjunction with the information presented earlier in Section 3.3.3.4. Pins 6, 7, 9, and 11 are not used in the current configuration of *SIMSAT*.

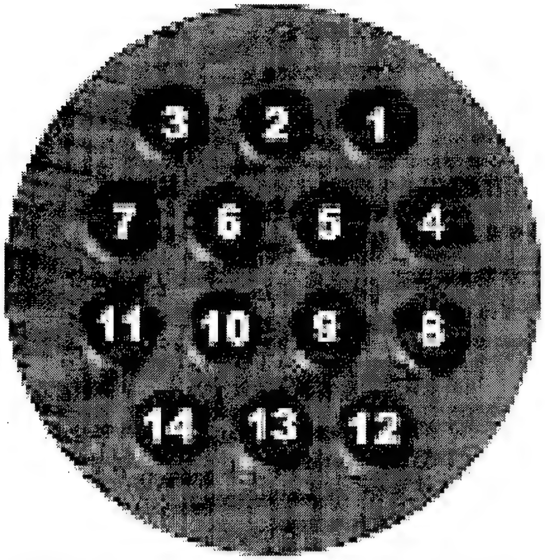


Figure 3.24 BL-3450 Pin Assignments

Table 3.8 BL-3450 Motor Pin Assignments

Pin Number	Parameter
1	Phase A
2	Phase B
3	Phase C
4	+5V
5	GND
6	+ Index Input
7	- Index Input
8	+ Encoder A Input
9	- Encoder A Input
10	+ Encoder B Input
11	- Encoder B Input
12	Hall Sensor 1
13	Hall Sensor 2
14	Hall Sensor 3

3.3.5 Momentum Wheel.

The momentum wheel stores angular momentum which can then be transferred to *SIMSAT* to maintain or change orientation.

Manufacturer: AFIT Fabrication Shop Bldg. 470

Point of Contact: Mr. Jan LaValley

Phone Number: (937) 255-2950

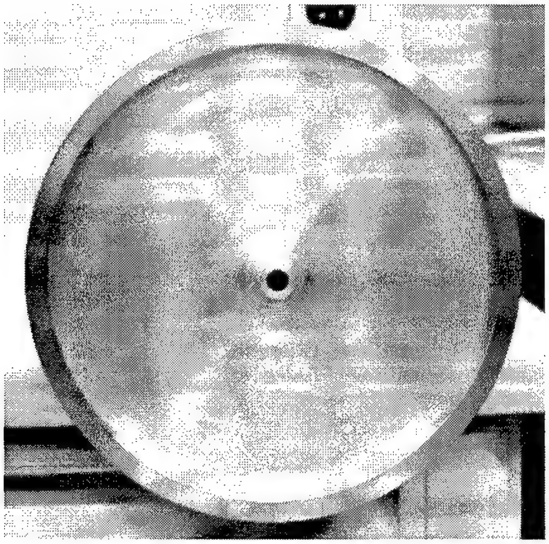


Figure 3.25 Steel Rim/Aluminum Disk Momentum Wheel

3.3.5.1 Characteristics.

Each wheel is comprised of a stainless steel hoop affixed to a thin aluminum disk. The wheel is designed to mount on the BL-3450's 3/8 inch motor shaft. The dimensions of the rim and the disk are listed in Table 3.9.

Table 3.9 Momentum Wheel Dimensions

Dimension	Value (in)	Value (cm)
Outer Diameter	8.625	21.9075
Hoop Width	0.375	0.9525
Hoop Thickness	1.1875	3.01625
Disk Thickness	0.25	0.635
Axle Hole Diameter	0.375	0.9525
Titanium Sleeve Diameter	0.625	1.5875
Aluminum Axle Outer Diameter	1.0	2.54

3.3.5.2 Mounting Considerations. The wheel is designed to mount on the BL-3450's 3/8 inch motor shaft. The end of each motor shaft is tapped. A screw is threaded into the tap and is secured with LOCTITE to prevent the momentum wheel from flying off the shaft. Additionally, the momentum wheel has two screws separated by 180 degrees within the aluminum axle wall. The screws press fit the wheel to the axle when tightened.

3.3.6 dSPACE AutoBox DS400. The AutoBox, shown in Figure 3.26, provides onboard control processing and signal consolidation.

Vendor: dSPACE Inc., 22260 Haggerty Road, Suite 120, Northville MI 48167
Manufacturer: dSPACE GmbH, Technologiepark 25, D-33100 Paderborn, Germany
Model No: DS400
Point of Contact: Mr. Vivek Moudgal, vmoudgal@dspaceinc.com
Phone Number: (248) 344-0096 x208
Web Site: www.dspace.de

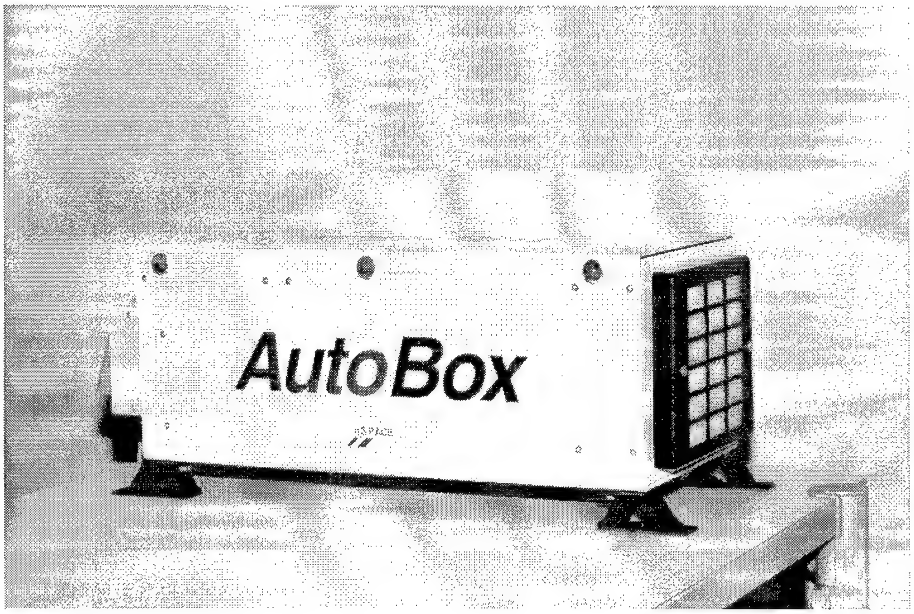


Figure 3.26 AutoBox

3.3.6.1 ***Characteristics.*** The AutoBox provides remote processing and signal consolidation. It is fully-compatible with the ground-based dSPACE software. The AutoBox supports 32 input and 32 output data channels. In addition to an Ethernet communications and power card, the AutoBox includes three dSPACE cards: the DS1003 DSP card, the DS2003 A/D card, and the DS2103 D/A card. Additional cards can be purchased from dSPACE if necessary.

3.3.6.2 ***Mounting Considerations.*** Although the manufacturer tested the AutoBox to operate under 5g vibrations from 5 to 2000 Hz, it is isolated from vibration as much as possible. Shock mounts are used to provide a base for the AutoBox mounting. A U-clamp is strapped over the AutoBox, with a polyurethane padding underneath to provide vibration isolation. Padded L-brackets provide support in the longitudinal direction. Figure 3.27 shows the AutoBox mounting plate.

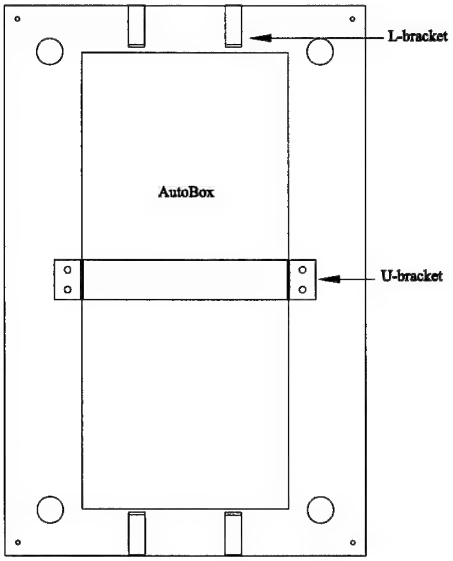


Figure 3.27 Autobox Mounting

3.3.6.3 Power Requirements/Interface.

The AutoBox operates under 8-36 VDC, with autoranging capability. Using an external DC power supply, laboratory testing has shown the AutoBox to use a constant 60W power consumption. At the operationally supplied voltage of 24 VDC, this corresponds to a 2.5A current.

3.3.6.4 Signal Requirements/Interface.

The AutoBox uses two 32-wire ribbon cables to interface with the channel interface board (see Section 3.3.9). One wire provides 32 input channels, and the other provides 32 output channels. For serial data (wireless modem), the AutoBox uses the RS-422-A high density sub-D connector. The wireless LAN connection uses a TRD450CR-series cross-wired cable to connect the DockLINK to the 10BaseT port of the AutoBox.

3.3.7 RadioLAN Wireless LAN.

The wireless LAN connection provides 10 Mbps Ethernet connectivity for the transfer of COCKPIT and TRACE data between the simulation PC and the onboard AutoBox. The ISA CardLINK with transceiver provides the offboard wireless station, whereas the DockLINK (shown in Figure 3.28) provides the onboard wireless station.

As the AFIT frequency manager, Mr. Kurt Gausel (AFIT/SC) approved use of the RadioLAN wireless system for the AFIT/ENY laboratories. He also forwarded the information to the 88ABW frequency manager.

Vendor: RadioLAN, 455 deGuine Drive, Suite D, Sunnyvale CA 94086
Model No: ISA CardLINK 101c, and DockLINK 408
Point of Contact: Andy at Technical Support
Phone Number (Tech Support): (408) 616-6333 or -6332
Phone Number (Sales): (408) 524-2600
Web Site: www.radiolan.com

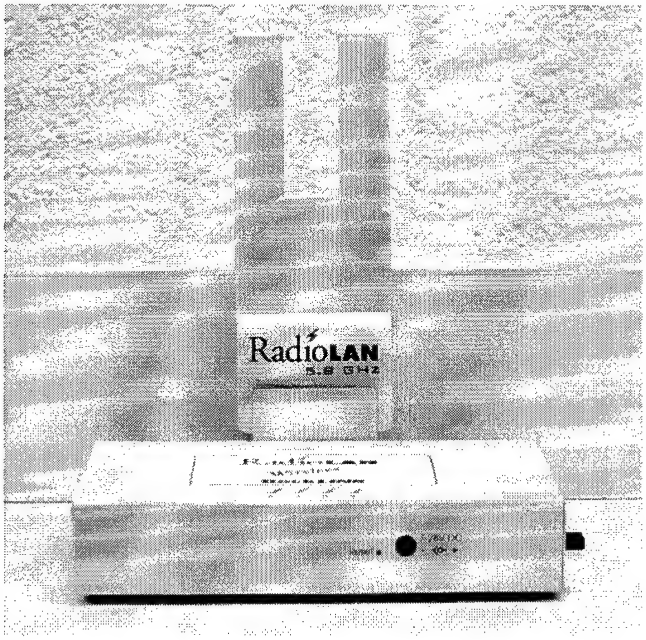


Figure 3.28 RadioLAN DockLINK with Transceiver

3.3.7.1 ***Characteristics.*** Table 3.10 lists the specifications of both the onboard and offboard wireless stations. Detailed installation and operation instructions are provided in the ISA CardLINK and DockLINK user guides, provided by RadioLAN. Dual omnidirectional antennas are used for RF transmission. The installation procedures indicate that the researcher should mount the transceiver parallel to the ground station transceiver. With *SIMSAT* in motion, this parallel alignment is not possible. Conversation with RadioLAN technical support (Andy) confirmed that in a short-range application such as this design, a rotating transceiver is not an issue.

3.3.7.2 ***Power Requirements/Interface.*** The onboard DockLINK's power cable is connected to the 12 VDC power bus bar.

3.3.7.3 ***Signal Requirements/Interface.*** The DockLINK uses the RJ-45 jack. A TRD450CR-series cross-wired cable connects the DockLINK to the 10BaseT port of the AutoBox.

Table 3.10 RadioLAN Product Data

Parameter	ISA CardLINK	DockLINK
Purchase Cost	$349	$799
Radio Frequency	5.8GHz ISM band	5.8GHz ISM band
Transceiver Weight	9.8oz (278g)	7.4oz (206g)
Unit Weight	4.6oz (130g)	22.3oz (632g)
Input Power	5V/12V	12V (or 110VAC)
Transmission Power	50mW peak	50mW peak
Media Access Protocol	CSMA/CA	CSMA/CA
Network Interface	16-bit ISA	RJ-45 jack
Warranty	1 yr (parts & labor)	1 yr (parts & labor)

3.3.8 Wireless Modem.

The wireless modem is used for the transfer of RealMotion data between the RealMotion PC and the AutoBox. *As of yet, AFIT has not purchased the wireless modem* . The Digital Wireless Corporation WIT2400E Developer's Kit is selected for future acquisition and integration. The RealMotion animation is a non-critical system element.

Vendor: Digital Wireless Corporation,

Model No: WIT2400 Developer's Kit

Point of Contact: Mr. Don Neas, dneas@digiwrls.com

Phone Number: (770) 564-5540 x2361

Web Site: www.digital-wireless.com

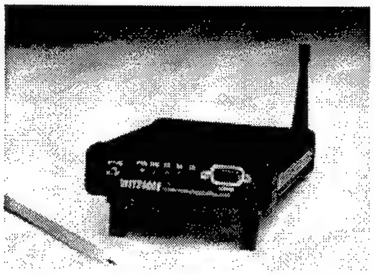


Figure 3.29 Digital Wireless Corporation WIT2400E Modem

3.3.8.1 ***Characteristics.*** The Developer's Kit includes two wireless modems (model WIT2400M), self-contained battery packs, battery charger, flow control indicators, dipole antennas, RS-232 interface, and configuration software. Table 3.11 lists some of the important parameters of the modem.

Table 3.11 WIT2400M Wireless Modem Characteristics

Parameter	Value
Dimensions	4.5" x 6" x 1.5"
Mass	1 kg (approx.)
Power Supply	AA NiCad Battery Pack (supplied)
Data Connector	DB-25
Interface Adapter	RS-232
RF Frequency	2.4 GHz
I/O Data Rate	Up to 115 Kbps
Transmitted Power	10 mW or 100 mW

3.3.8.2 ***Mounting Considerations.*** As a small, lightweight item, the researcher can mount the wireless modem anywhere convenient. One option is to mount the modem near the AutoBox to minimize wiring. This option, however, will increase the overall system mass since more mass is added to the counterweight mechanism. The other option is to place the modem near the counterweight plate. This will reduce overall system mass and minimize signal interference between the modem and the Radio LAN.

3.3.8.3 ***Power Requirements/Interface.*** The AA NiCad battery pack and charger is supplied with the Developer's Kit. Thus, no interface with the onboard power supply is required.

3.3.8.4 ***Signal Requirements/Interface.*** The RS-232 interface connects to the RS-422-A high density sub-D connector on the AutoBox. Thus, a converter is required. Black Box Corporation (www.blackbox.com) produces a non-powered RS-232 to RS-422 converter for a DB-25 data connector - part number IC470A (male or female). Information on the feasibility of this connector is yet undetermined. Detailed signal pinout of the WIT2400M is displayed on the Digital Wireless website (www.digital-wireless.com/wit24002.html).

3.3.9 Channel Interface Board.

The interface board (see Figure 3.30) provides a 32 input/32 output data channel interface between onboard components and the AutoBox. This interface board consists of connector ports for the input and output signals to each component. Four additional input and four output ports are available to accomodate future research needs.

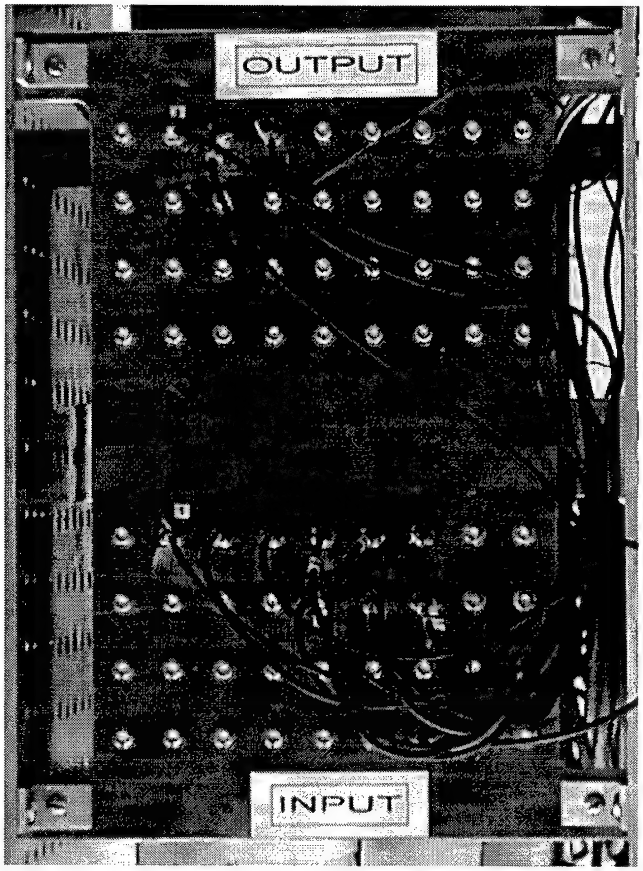


Figure 3.30 Channel Interface Board

3.3.9.1 Characteristics. RG174/U data cables are used to connect *SIMSAT*'s operational components to the channel interface board. The first column of input and output ports are not wired into the 32-wire ribbon cables leading to the AutoBox. Table 3.12 identifies each data cable and the component it connects to. The appropriate port to attach each data cable to is determined by the experimental model.

Table 3.12 Channel Interface Board Data Cables

Number	Component
1	Velocity Monitor Input from Amp #1
2	Current Monitor Input from Amp #1
3	Velocity Monitor Input from Amp #2
4	Current Monitor Input from Amp #2
5	Velocity Monitor Input from Amp #3
6	Current Monitor Input from Amp #3
7	Voltage Output to Amp #1
8	Voltage Output to Amp #2
9	Voltage Output to Amp #3
13	Pitch Rate from Gyro
14	Roll Rate from Gyro
15	Yaw Rate from Gyro
16	Fore and Aft Acceleration
17	Lateral Acceleration
18	Vertical Acceleration

3.3.9.2 Mounting Considerations. The channel interface board is mounted along side of the AutoBox. One end is fastened to the AutoBox mounting plate while the other end is fastened to the payload mounting plate. This configuration minimizes the amount of 32-wire ribbon cable needed to connect the interface board to the AutoBox. This configuration also reduces the length of payload data cables.

3.3.9.3 Power Requirements/Interface. The channel interfaces are non-powered connectors.

3.3.9.4 Signal Requirements/Interface. Inputs and outputs from all components should use ±5V or ±10V analog signals.

3.3.10 Power Architecture. Three 12V batteries wired in series provide the required power to operate the functional components of *SIMSAT* and experimental hardware. A power bus bar, shown in Figure 3.31, is used to provide 12V, 24V, and 36V bus connections.

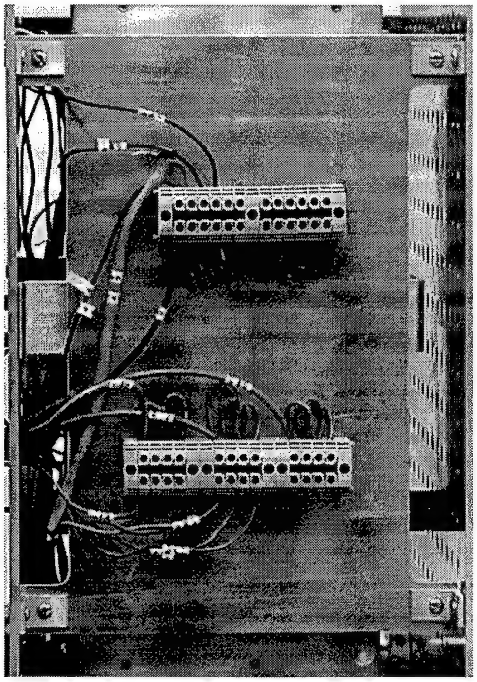


Figure 3.31 Power Bus Bar

The top bus bar provides a common ground for the functional components and also completes the circuit of the batteries. The bottom bus bar allows access to 12V, 24V, and 36V. Another power cable runs through the hollow mounting shaft to another power bus bar mounted to the amplifier mounting plate. This bus bar supplies 36V power to the three amplifiers.

Total power capacity of the baseline *SIMSAT* is 35.1Ahr at the 0.5C (1.3 hr) discharge rate, or 29.7Ahr at the 1.0C (33-min) discharge rate. A total available continuous current of 27A is possible at the 0.5C rate, or 54A at the 1.0C rate. *SIMSAT* can draw additional current for brief periods of time but this increase in current is not sustainable and may decrease lifetime battery performance.

The researcher selected various wire gauges to ensure an adequate level of protection against overheating. Nominal wire gauges for *SIMSAT* components are shown in Table 3.13.

Table 3.13 Nominal Wire Gauges

Component	Current (A)	Voltage (V)	Power (W)	Min Wire Gauge
AutoBox	2.5	24	60	18
Motor/Amp 1	20	36	720	12/14
Motor/Amp 2	20	36	720	12/14
Motor/Amp 3	20	36	720	12/14
Receiver/Transmitter	0.5	12	6	27
Gyro	1.78	28	50	18

3.3.10.1 Mounting Considerations. The power bus bar mounting board is mounted along side of the AutoBox, opposite from the channel interface board. One end is fastened to the AutoBox mounting plate while the other end is fastened to the payload mounting plate. This configuration offers close access to the three batteries and offers easy accessibility to power experimental hardware.

Three toggle switches are mounted across from the undervoltage alarm (see Figure 3.32). These switches provide a main location for turning on and off the functional components of *SIMSAT*.

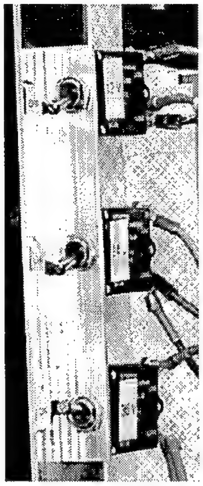


Figure 3.32 Power Toggle Switches

3.3.11 Batteries. Vendor Information

Vendor: Power-Sonic Corporation
9163 Siempre Viva Road
San Diego, CA 92173
Model No: PS-12180 F2
Point of Contact: Mr. Javad (last name unknown)
Phone Number: (619) 661-2020
Email: javad@power-sonic.com

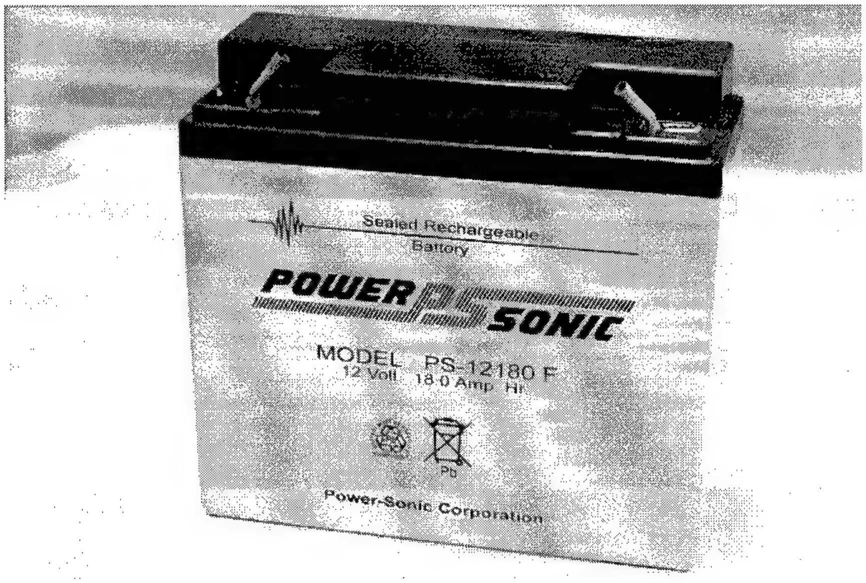


Figure 3.33 Power-Sonic PS-12180 Sealed Lead-Acid Battery

3.3.11.1 Characteristics. *SIMSAT* power is provided from three rechargeable Power-Sonic PS-12180 sealed lead-acid batteries, shown in Figure 3.33. The sealed design of the PS-12180 allows for unrestricted operation under all possible *SIMSAT* orientations. Each battery is nominally rated with an 18-Ahr capacity at a DC operating voltage of 12V. The battery case is made of non-conductive polystyrene for high impact resistance. Battery mass is approximately 5.9kg each.

In the event of excessive gas buildup within a battery, a one-way neoprene-rubber relief valve provides a safety mechanism to ensure safe depressurization without case rupture occurring. Vent release pressure is between 2-6 psi. Each battery is mounted within an aluminum enclosure that acts as a mounting interface to the *SIMSAT* truss structure, as well as providing additional safety in the event of a battery leak. This arrangement also allows for ease of battery changeout between experiments.

3.3.11.2 Battery Operation-Practical Considerations. The following guidelines are extracted from the Power-Sonic Sealed Lead-Acid Battery Technical Handbook:

- Continuous over- or undercharging is the worst enemy for battery life. Be cognizant of the appropriate charge times for a battery. Use the appropriate Power-Sonic charger. Three PSC-122000A chargers are available to recharge all three batteries simultaneously. Alligator clips allow recharging of the batteries without removing the batteries from their mounting boxes.
- Do not store batteries in a discharged state or at elevated temperatures. Avoid exposing batteries to heat! Service life is shortened considerably at ambient temperatures above 30 degrees Celsius.
- Power-Sonic recommends that unused batteries be charged 6-9 months after receipt to prevent permanent loss of capacity due to sulfation. Store batteries at 10 degrees Celsius or less.
- Recharge time depends on the depth of the preceding discharge and the output current of the charger. To determine the approximate recharge time of a fully discharged battery, divide the battery's capacity (in Ahrs) by the rated output of the charger (in Amps) and multiply the resulting number of hours by 1.75 to compensate for the declining output current during charge. If the number of Ahrs discharged from the battery is known, use it instead of the battery's capacity when making the calculation.

- It is not recommended to attempt rapid (6-7 hr) recharge of a Power-Sonic battery. Too high a charge current can destroy a battery within hours.
- Never charge or discharge a battery in an airtight enclosure. Do not place batteries in close proximity to sparks or open flame. Do not charge batteries in an inverted position.
- Do not expose Power-Sonic batteries to organic solvents or adhesives. Damage to the battery case may result.
- Do not attempt to disassemble batteries. Contact with sulfuric acid is dangerous and may cause chemical burns. In the event of exposure, wash skin or clothes with liberal amounts of water. Dispose batteries as hazardous waste at the end of their service lives.

3.3.12 Undervoltage Alarm. *SIMSAT* battery discharge levels are monitored during *SIMSAT* experiments. The researcher must terminate operation of *SIMSAT* prior to battery power falling below acceptable minimums. Premature shutdown due to inadequate power margins could result in the uncontrolled dumping of system momentum and catastrophic *SIMSAT* tumbling.

Warning of a low power condition is provided by a Macromatic VMP024D voltage monitoring relay wired across two of the three *SIMSAT* batteries (see Figure 3.34). The relay de-energizes when the monitored voltage drops below 23.3V, corresponding to a remaining system battery capacity of approximately 10%. De-energizing the VMP024D closes a circuit wired to a sonoalarm (audible tone buzzer). The operator should immediately begin the orderly termination of *SIMSAT* operations upon activation of the sonoalarm.

A future method of implementing power monitoring capabilities would rely upon the data collection and telemetry subsystem already onboard *SIMSAT*. The researcher can monitor voltage and current (across each of the batteries individually, or collectively) using voltmeters and ammeters. Data input channels routed to AutoBox would collect telemetry from these sensors, which in turn would transmit the telemetry stream to the

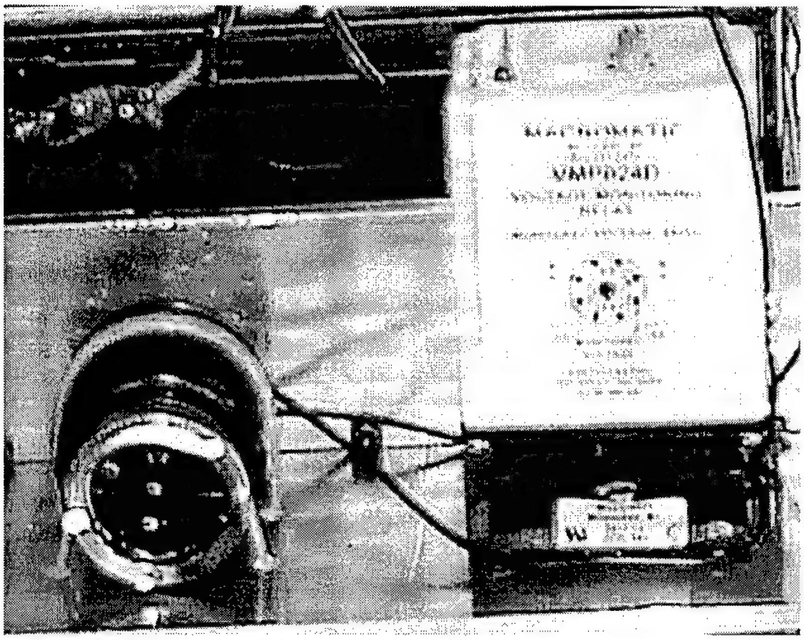


Figure 3.34 Macromatic VMP024D Voltage Monitoring Relay

ground station. The current *SIMSAT* software code supports future implementation of power subsystem telemetry.

3.4 SIMSAT *Software*

3.4.1 Software Overview. Several software programs are used to execute *SIMSAT* experiments. These programs are loaded on the ground station PC and the RealMotion PC PC located in the *SIMSAT* lab in building 640, room 146. The six main software programs include:

- AutoCAD
- 3D Studio VIZ
- MATLAB
- SIMULINK
- dSPACE
- RealMotion PC

3.4.2 AutoCAD.

AutoCAD Release 14.01 is a powerful tool for supporting the *SIMSAT* system. With AutoCAD, individual components are drawn, dimensioned, and printed for delivery to the Fabrication Shop. The individual drawings combine to form a composite picture of the overall *SIMSAT* structure. Although the drawings are two-dimensional images on the computer monitor, AutoCAD is capable of treating the drawings as 3D objects. As a result, AutoCAD allows the user to view an object from different perspectives, rotate it, determine its volume, locate its centroid, and estimate its inertia tensor.

Most drawings in AutoCAD are saved as "*.dwg" files. However, objects drawn with simple polygons or polygon meshes may be saved as "*.dxf" files. Only objects saved as "*.dxf" files are compatible with dSPACE's RealMotion software. AutoCAD is installed on the RealMotion PC PC in room 146.

3.4.3 3D Studio VIZ.

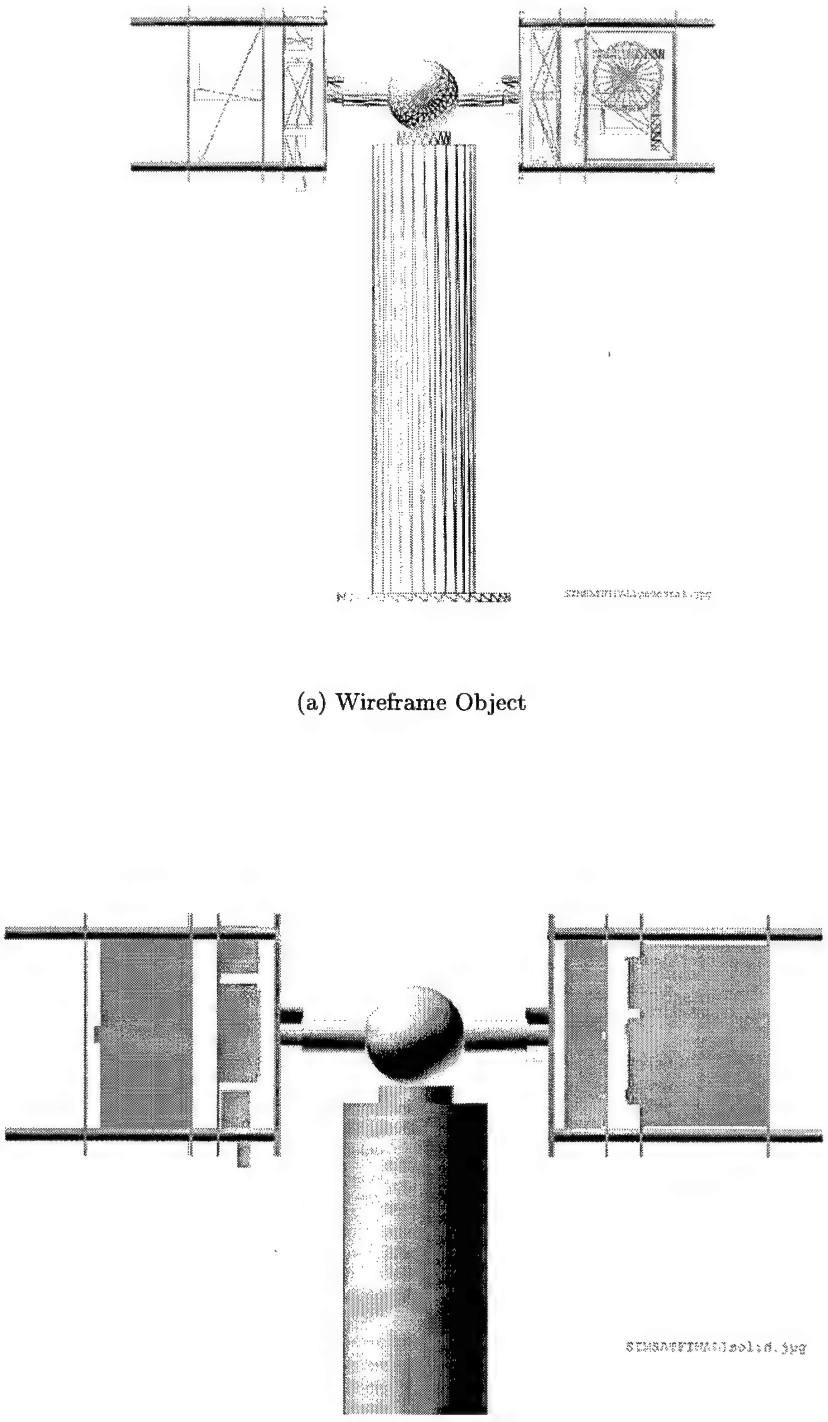
3D Studio VIZ R2 is installed on the same computer as AutoCAD. As the name implies, 3D Studio VIZ allows enhanced visualization of three-dimensional objects. Objects are drawn directly in 3D Studio VIZ or imported from AutoCAD drawing files. In 3D Studio VIZ, objects are displayed as either solid forms or as wireframes. With a click of the mouse button, objects are translated or rotated. A series of rotations and/or translations combine to form animation. A scene is saved into a "*.jpg" format. A program such as Paint Shop Pro Version 4.12 will open and print the scene (see Figure 3.35).

3.4.4 MATLAB Files.

MATLAB 5.3.0 R11 is loaded on the ground station PC in the *SIMSAT* lab. The computer contains MATLAB files related to the equations of motion (eom) used to model *SIMSAT*. A printout of these files is located in Appendix A. A brief description of each file is given below.

- **"inerbal.m"**–utilizes the functions of AutoCAD to develop the composite *SIMSAT* inertia matrix with respect to *SIMSAT*'s center of mass. This program is paired with

"SIMSATFINAL1nopayldpltcwmechBALANCE.dwg" and
"SIMSATFINAL1withobjectgrouping&8RODS.dwg"



(a) Wireframe Object

(b) Solid Form Object

Figure 3.35 3D Studio VIZ Rendering

(or other yet-to-be-created *SIMSAT* drawings) in AutoCAD. The "inerbal.m" program also calculates the composite center of mass of *SIMSAT* and aids the user with static balancing (i.e., attempting to make the position vector FROM the origin at the center of the sphere TO the composite center of mass equal to [0; 0; 0]). "inerbal.m" will prompt the user for the horizontal distance between the base plate center of mass (c.o.m.) and the c.o.m. of a mounting plate (excluding components). When calculating the position vectors of the components (gyro, motors, batteries, amplifiers, etc.), "inerbal.m" assumes the researcher has not changed the positions of the components relative to their respective mounting plates from the baseline design. This program also determines the inertia of each momentum wheel (using the 8 & 5/8" steel hoop and 1/4" aluminum disk baseline design) because the wheel inertias are required for the SIMULINK and MATLAB motion simulations. "inerbal.m" concludes by calculating the inverse matrix of the first four terms (these terms do not vary with time) of the equations of motion (eom).

- "**quikiner.m**"– allows the user to directly enter the composite *SIMSAT* inertia tensor in kg*m^2 units. This program assumes *SIMSAT* is already balanced (i.e., composite center of mass very close to [0; 0; 0] in the body-fixed "b" frame). It is also assumed the user knows the composite system inertia matrix about the *SIMSAT* center of mass and the position vectors (in cm) FROM the center of the sphere TO the center of mass for each momentum wheel. After the user enters the composite inertia tensor and wheel position vectors, this program calculates the inertia of each momentum wheel (using the 8 &5/8" steel hoop and 1/4" aluminum disk baseline design) because the wheel inertias are used by the SIMULINK and MATLAB motion simulations. "quikiner.m" concludes by calculating the inverse matrix of the first four terms (these terms do not vary with time) of the equations of motion (eom).

- "**simcloop3.m**" The user runs "simcloop3.m" AFTER running a program (such as "inerbal.m" or "quikiner.m") to generate the required inertia matrices. This program contains the block diagram algebra necessary to implement the closed-loop control diagram shown in Chapter IV. The user is allowed to vary matrix gains to influence *SIMSAT* motion response. After calling "domega2.m" to numerically integrate the equations of motion, "simcloop3.m" provides output graphs of *SIMSAT* attitude angles, angular velocities, and momentum wheel speeds vs. time.

- "**domega2.m**"– is not an executable file; it is a MATLAB function that is called by "simcloop3.m". This program contains the *SIMSAT* equations of motion and supports the MATLAB "ode45" routine that numerically integrates the *SIMSAT* state vector.

- "**solver2play.m**"–The user runs this program AFTER running a program (such as "inerbal.m" or "quikiner.m") to generate the required inertia matrices. Since the *SIMSAT* composite inertia matrix contains products of inertia, some motion coupling should exist between the roll, pitch, and yaw axes. "solver2play.m" illustrates this coupling by simulating *SIMSAT* motion for an **open-loop**, "bang-bang" 10 second yaw maneuver. Because *SIMSAT* is operated mostly in a **closed-loop** control mode, "solver2play.m" is rarely used. After calling "domega.m" to numerically integrate the equations of motion, "solver2play.m" provides output graphs of *SIMSAT* attitude angles, angular velocities, and momentum wheel speeds vs. time.

- "**domega.m**"– is not an executable file; it is a MATLAB function that is called by "solver2play.m". This program contains the *SIMSAT* equations of motion and supports the MATLAB "ode45" routine that numerically integrates the *SIMSAT* state vector.

3.4.5 SIMULINK Command and Control Software.

SIMULINK is a graphical user interface (GUI) add-on to MATLAB. This software tool allows dynamic system simulation for *SIMSAT*. It is the primary software environment used for creating the *SIMSAT* command and control code which is compiled and downloaded to the AutoBox. SIMULINK is based on user-friendly, block-diagram coding. It provides excellent modularity for future experimental additions and control law modifications. Examples of the SIMULINK code blocks available for use with the baseline *SIMSAT* software architecture are shown in Figure 3.36.

SIMULINK, Version 3.4, is installed on the "ground control" PC located in the *SIMSAT* lab (room 146). This computer is connected to the AutoBox using the RadioLAN. This allows SIMULINK code compilation and download directly to the AutoBox for non-real-world *SIMSAT* simulations.

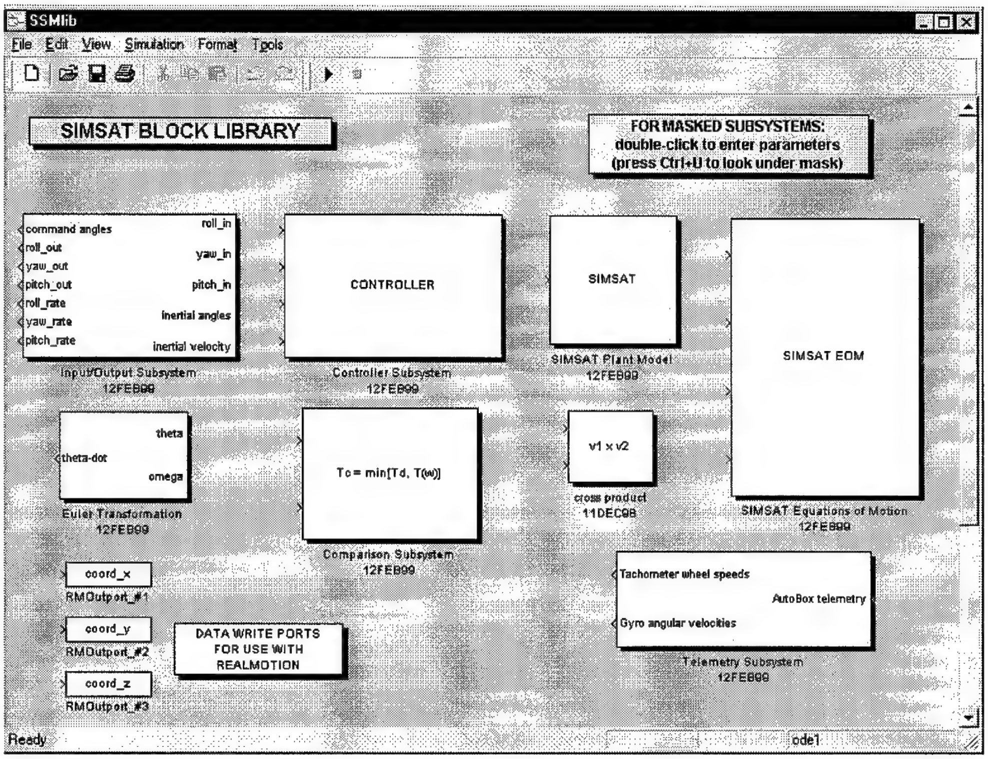
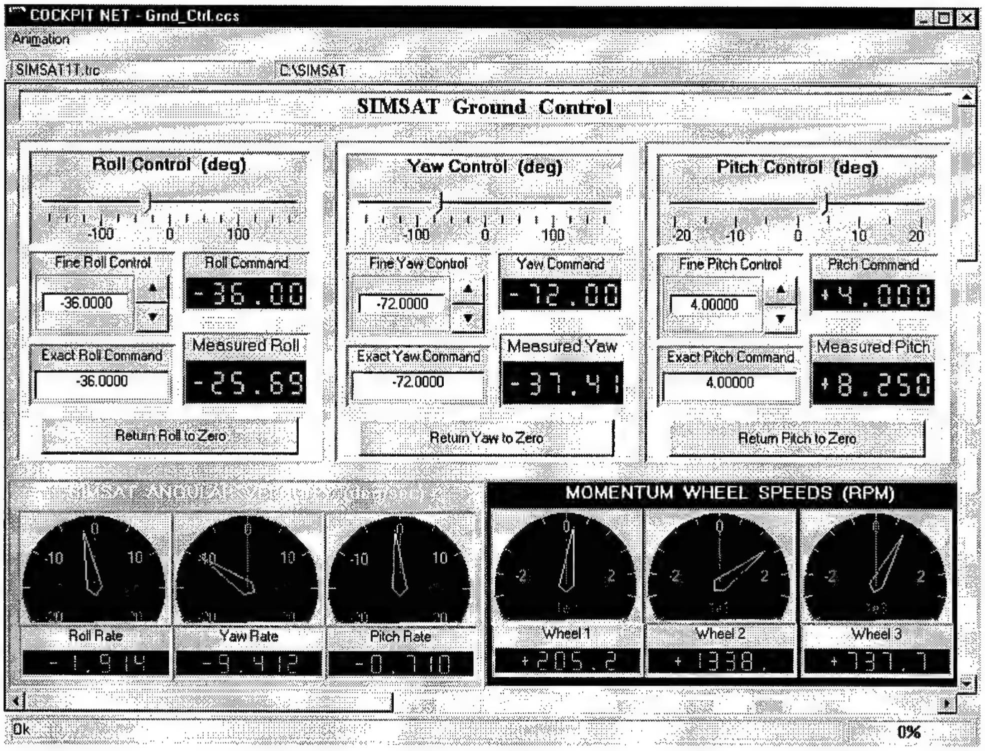


Figure 3.36 *SIMSAT* SIMULINK Block Library

The Real-Time Workshop (RTW), for use with SIMULINK 3.4, and Real-Time Interface (RTI), Version 3.0, are software tools that work with SIMULINK and the dSPACE system to compile SIMULINK code into C-code and download it to the DS1003 card in the AutoBox. Their seamless incorporation with SIMULINK provides a simple compilation process. For a description of SIMULINK's operational use with *SIMSAT* refer to Section 4.3.

3.4.6 dSPACE Experimentation Software. dSPACE Version 2.1 is loaded on the ground station PC in the *SIMSAT* lab. This software provides a visual experiment platform for executing commands to and receiving data from *SIMSAT*. The COCKPIT instrument panel and the real-time TRACE module are both incorporated into the dSPACE software. For a description of dSPACE's operational use with *SIMSAT* refer to Section 4.4.

3.4.6.1 COCKPIT Instrument Panel. The COCKPIT application provides a Graphical User Interface for real-time command and control of *SIMSAT* operations. Various control and display instruments, such as slide bars, numeric inputs, and gauges, create a user-friendly link to *SIMSAT* motion. Figure 3.37 illustrates the User Interface for the baseline *SIMSAT* system.



COCKPIT NET - Grnd_Ctrl.ccs

SIMSAT1T.trc C:\SIMSAT

SIMSAT Ground Control

Fine Roll Control: -36.0000
Roll Command: -36.00
Exact Roll Command: -36.0000
Measured Roll: -25.69

Fine Yaw Control: -72.0000
Yaw Command: -72.00
Exact Yaw Command: -72.0000
Measured Yaw: -37.41

Fine Pitch Control: 4.00000
Pitch Command: +4.000
Exact Pitch Command: 4.00000
Measured Pitch: +8.250

Roll Rate: -1.914
Yaw Rate: -9.412
Pitch Rate: -0.710

Wheel 1: +205.2
Wheel 2: +1338.
Wheel 3: +737.7

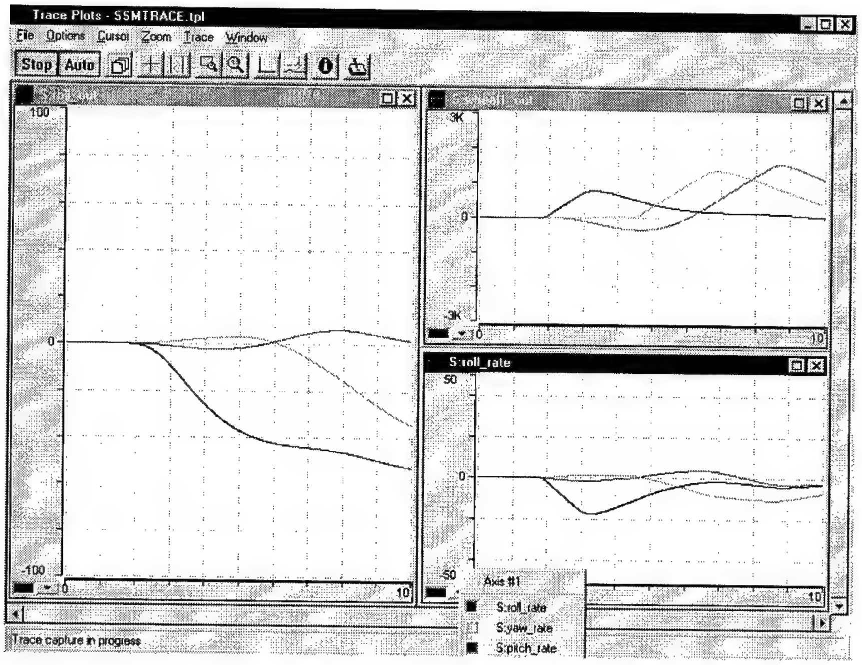
Ok 0%

Figure 3.37 COCKPIT Command and Control Graphical User Interface

3.4.6.2 Real-Time TRACE *Module.* The TRACE application provides a continuously updating display of *SIMSAT* telemetry. It is used to plot the time histories of *SIMSAT* roll, yaw, and pitch angular positions and velocities, as well as the speeds of the three momentum wheels. Figure 3.38 shows the baseline *SIMSAT* telemetry display screen.

3.4.7 RealMotion PC 3-D Animation Tool.

The RealMotion PC application provides a real-time graphical representation of *SIMSAT* motion. It connects a 3-D geometric model of *SIMSAT*, developed using AutoCAD, with angular position



Trace capture in progress

Figure 3.38 TRACE Real-Time Telemetry Display

variables in the *SIMSAT* on-board code. Figure 3.39 presents the geometric model used for real-time graphical representation of *SIMSAT* motion. RealMotion PC, Version 1.0.3, is installed on the "RealMotion" PC in the *SIMSAT* lab.

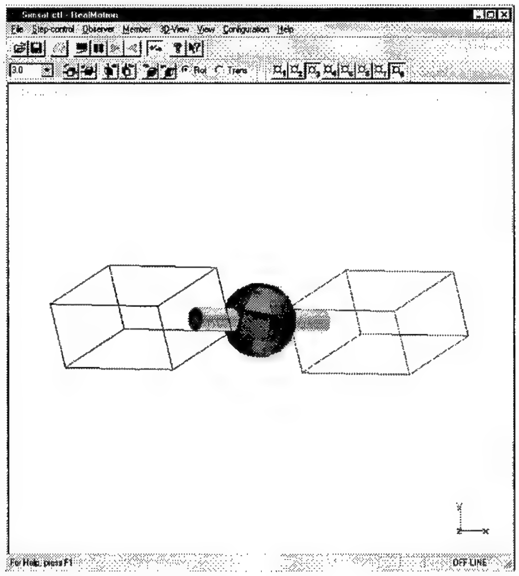


Figure 3.39 RealMotion PC Real-Time 3-D Animation Display

3.5 Summary

The experimental model of *SIMSAT* is comprised of many components. Components such as the air bearing assembly and the mounting plates provide overall structure. The AutoBox and the Humphrey CF-75 Series Axis Rate Gyro are some of the functional components used in system operation. Software programs, such as SIMULINK and dSPACE, allow the researcher to interact with the performance of *SIMSAT*. All of the components discussed in this chapter come together to form an ideal experimental satellite testbed as well as a visual educational tool.

IV. Experimental Results

4.1 Overview

Upon completion of the construction phase of *SIMSAT*, the author conducted experimental testing of the system to fine-tune operational procedures. This chapter begins with a discussion of the procedures to be followed by future researchers to connect the ground station PC to the AutoBox. Next, a detailed explanation of the SIMULINK model design used to operate *SIMSAT* is provided. This section is followed by a description of the graphical user interface the researcher uses to transmit and receive telemetry data. The chapter concludes with a section covering the multimedia lesson plan and lab experiment developed to demonstrate basic satellite dynamics.

4.2 Ground Station PC and AutoBox Connection

The ground station PC is configured to communicate with the AutoBox through the use of the wireless RadioLAN. If the researcher wishes to use a cable to direct connect the two computers, or if the hardware settings on the ground station PC are deleted, then the following steps will assist the researcher in properly configuring the system.

1. Select on Control Panel on the ground station PC
2. Select the Network Icon
3. Under the configuration tab, scroll down to TCP/IP → RadioLan 10/ISA Radionet Interface Driver
4. Select the properties box
5. Under the IP Address tab, select Specify an IP address
6. Enter in the IP Address block 192.100.100.1
7. Enter in the Subnet Mask block 255.255.255.0
8. Under the DNS Configuration tab, select Disable DNS

9. Select OK

Repeat steps 3-9 for TCP/IP → 3Com EtherLink XL TPC 10 Mb Ethernet NIC (3C900B-TPC). This procedure configures the hardware settings on the ground station PC.

After configuring the hardware settings on the ground station PC, the researcher must decide to use either the RadioLAN or direct cable connection (Ethernet). The Ethernet driver is listed before the RadioLAN driver in the device manager of the ground station PC. Because of this configuration, the researcher may experience difficulty in using the desired connection. The following list provides information on troubleshooting this problem:

1. Select on Control Panel on the ground station PC
2. Select the System icon
3. Select the Device Manager tab
4. Select Network adapters
5. Select 3Com EtherLink XL TPC 10 Mb Ethernet NIC (3C900B-TPC)
    - To use the Ethernet connection ensure Disable in this hardware profile block is not checked
    - To use the RadioLAN connection ensure Disable in this hardware profile block is checked
6. Click OK
7. Restart the computer

To determine if the ground station PC and the AutoBox are connected, the researcher will open up a DOS prompt window on the ground station PC and type ping 192.100.100.98. This command will send four packets of information to the AutoBox to determine connection status. If the packets are sent, then the response time is displayed. Request time out messages are received if the connection fails. If these messages are displayed, ping the AutoBox a second time. If this still fails, then the researcher must recheck the connection procedures.

Once the researcher has successfully connected the ground station PC to the AutoBox, *SIMSAT* is ready to transmit and receive information. The two programs which send information to *SIMSAT* are MATLAB and dSPACE Control Desk. Important precautions to connecting these two programs to *SIMSAT* are presented in Sections 4.3.4 and 4.4 respectively. Even though the ground station PC and the AutoBox are successfully connected, the researcher may experience difficulty in maintaining this connection. If the connection is lost accomplish the following procedure:

1. Ensure *SIMSAT*'s RadioLan antenna's indicator is blinking
2. Ensure the link light on the DOCKLINK is illuminated
3. Power down the AutoBox and the ground station PC
4. Disconnect and reconnect the data line between the AutoBox and the DOCKLINK
5. Power up the AutoBox
6. Wait for the AutoBox to fully boot up (three repetitive beeps sound)
7. Power up the ground station PC
8. Open up a DOS window on the ground station PC
9. Enter Ping 192.100.100.98
10. Check responses for reply messages
11. A failed connection results in 'request timed out' messages
12. If a failed message is received, ping the system one more time
13. A successful reply message looks like 'Reply from 192.100.100.98: bytes = 32 time = 1ms TTL = 64'
14. If the researcher receives both failed and successful messages, continue to ping the system until four successful messages are received

4.3 SIMULINK Model Design

The researcher uses SIMULINK to design the block diagrams which operate *SIMSAT*. These block diagrams contain the dynamic equations and stabilization control equations discussed in Chapter II. The SIMULINK model links the dSPACE Control Desk with the AutoBox. This link facilitates telemetry data transmition and reception between the ground station PC and *SIMSAT.*

4.3.1 Top Level Architecture.

The top level SIMULINK model is shown in Figure 4.1. The model is located on the SIMSAT 2000 Zip disk under the filename simsat1b.mdl. The command in-ports receive roll, yaw, and pitch commands from dSPACE Control Desk. This information is sent into the controller subsystem where it is combined with inertial orientation, measured angular velocities, and measured momentum wheel rates (see Section 4.3.2). This block outputs desired momentum wheel speeds to achieve desired orientation. These wheel speeds are displayed on the dSPACE Control Desk. The wheel speeds have units of revolutions per minute (RPMs). The AutoBox inputs and outputs voltages so the wheel speeds are converted from RPMs to volts. The author performed static testing of each motor to determine the conversion rates from RPMs to input voltage (see Section 4.3.3). Section 4.3.4 discusses how the telemetry signals are sent to the AutoBox.

The telemetry output from the AutoBox consists of measured momentum wheel rates and measured angular velocities. This information is sent into the telemetry subsystem where unit conversions occur (see Section 4.3.5). The measured wheel rate and measured angular velocities are fed back into the controller subsystem. In addition, the Euler transformation block iterates the angular velocities to calculate current inertial orientation of *SIMSAT* (see Section 4.3.6). The telemetry out-port blocks send roll, yaw, and pitch data to dSPACE Control Desk and also to the RealMotion PC. Roll, yaw, and pitch rates are also downloaded to dSPACE Control Desk where they are displayed for the researcher.

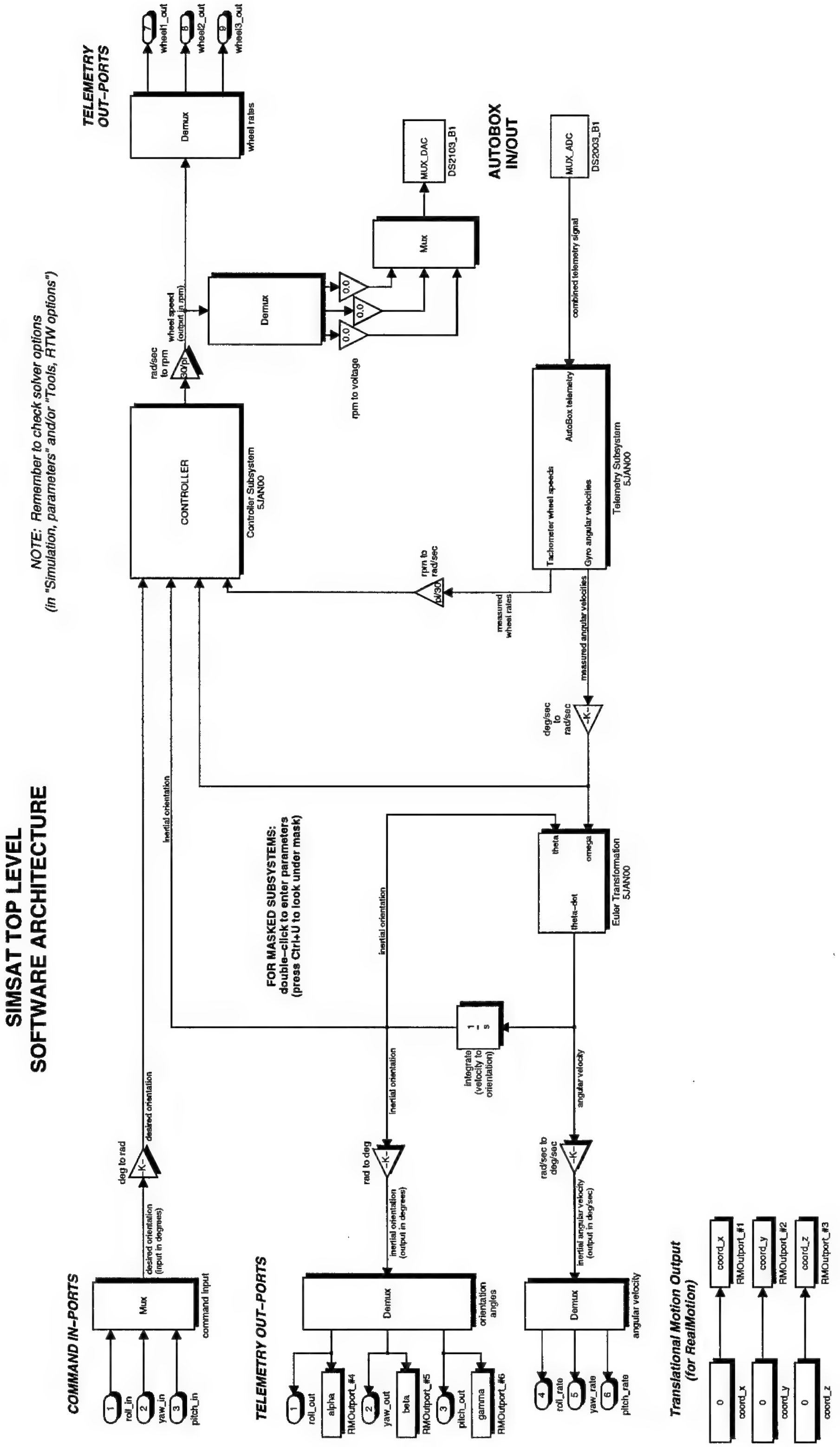


Figure 4.1 SIMULINK Model–Top Level Architecture

*4.3.2 **Controller Subsystem.*** Figure 4.2 is the controller subsystem for *SIMSAT*'s SIMULINK model. This block calculates the desired torque required to obtain commanded orientation. The momentum wheel controller equations presented in Section 2.5 are contained here. Figure 4.3 shows the values for the variables K1, K2, K3, and Iw. K1 represents position gain and K2 represents rate gains. K3 is a gain applied to the PD controller equation to allow the researcher to modify system response times without recalculating position and rate gains. Iw is the inertia of the momentum wheels.

The PD controller provides the desired torque required for *SIMSAT* to obtain the commanded orientation. However, momentum wheel speed saturation may prevent the system from producing the desired torque. The comparison subsystem block, shown in Figure 4.5, compares the desired control torque to the torque which the momentum wheels can actually produce. The minimum value of the two torques is selected at the control torque output.

The measured wheel rates are compared to the maximum wheel rate. If a momentum wheel is already rotating at its maximum rate, then a value of zero is generated for the control torque. If the momentum wheel has not reached its saturation limit, then its rate is entered into the torque equation (see Figure 4.4).

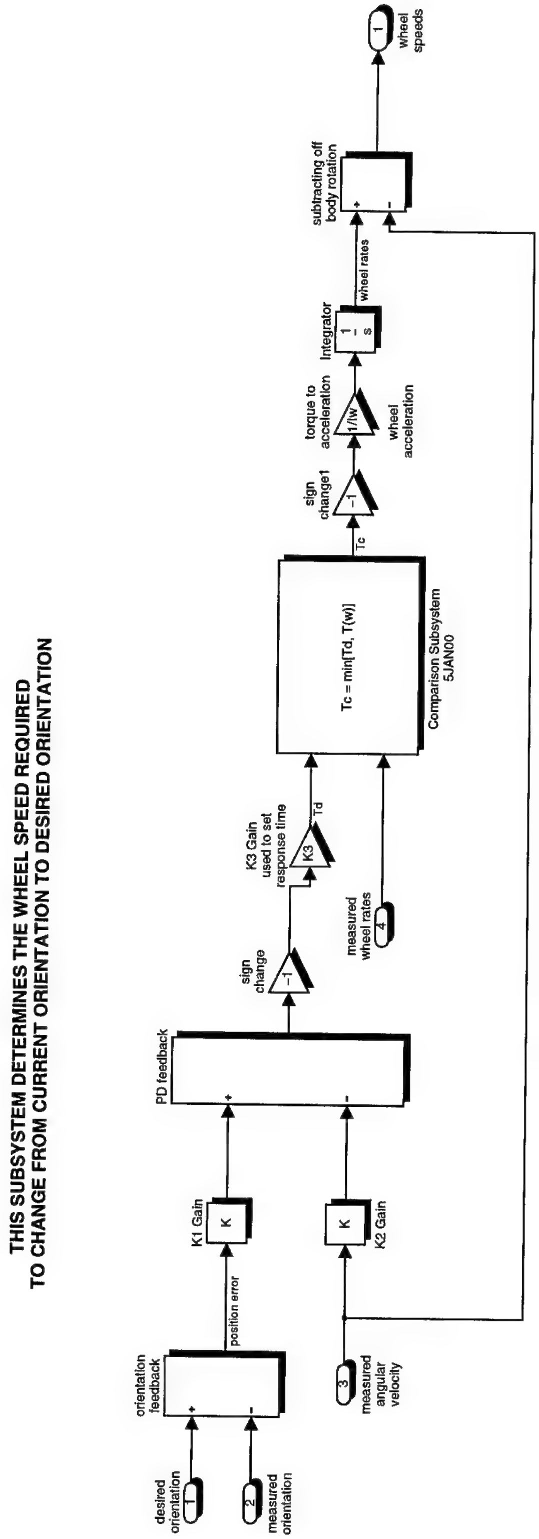
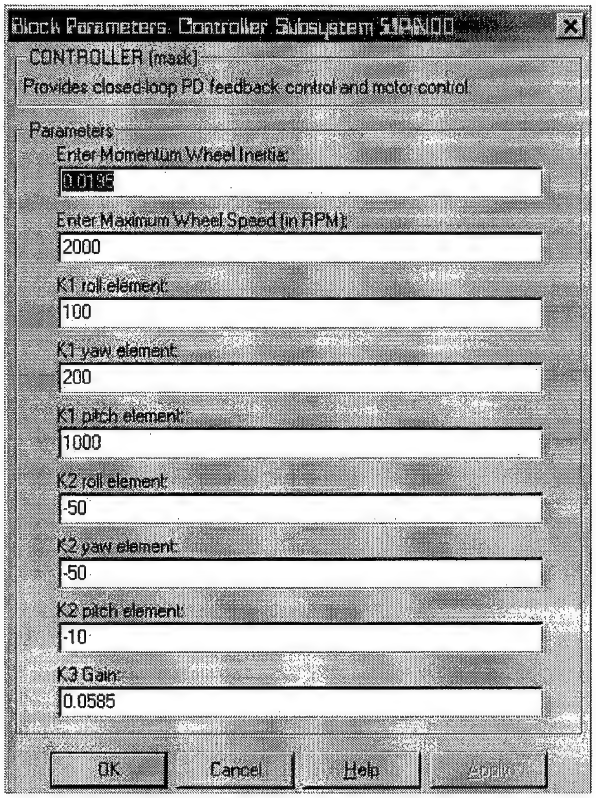


Figure 4.2 SIMULINK Model–Controller Subsystem



Block Parameters: Controller Subsystem SUPAN00

CONTROLLER (mask)

Provides closed-loop PD feedback control and motor control.

Parameters

Enter Momentum Wheel Inertia:
0.0195

Enter Maximum Wheel Speed (in RPM):
2000

K1 roll element:
100

K1 yaw element:
200

K1 pitch element:
1000

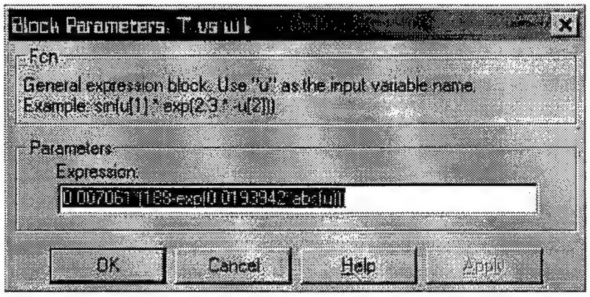
K2 roll element:
-50

K2 yaw element:
-50

K2 pitch element:
-10

K3 Gain:
0.0585

Figure 4.3 Controller Subsystem Variable Inputs



Block Parameters: T vs w k

Fcn

General expression block. Use "u" as the input variable name.
Example: sin(u[1] * exp(2.3 * -u[2]))

Parameters

Expression:
0.007061*(188-exp(0.0193942*abs(u)))

Figure 4.4 Motor Torque Equation Input Block

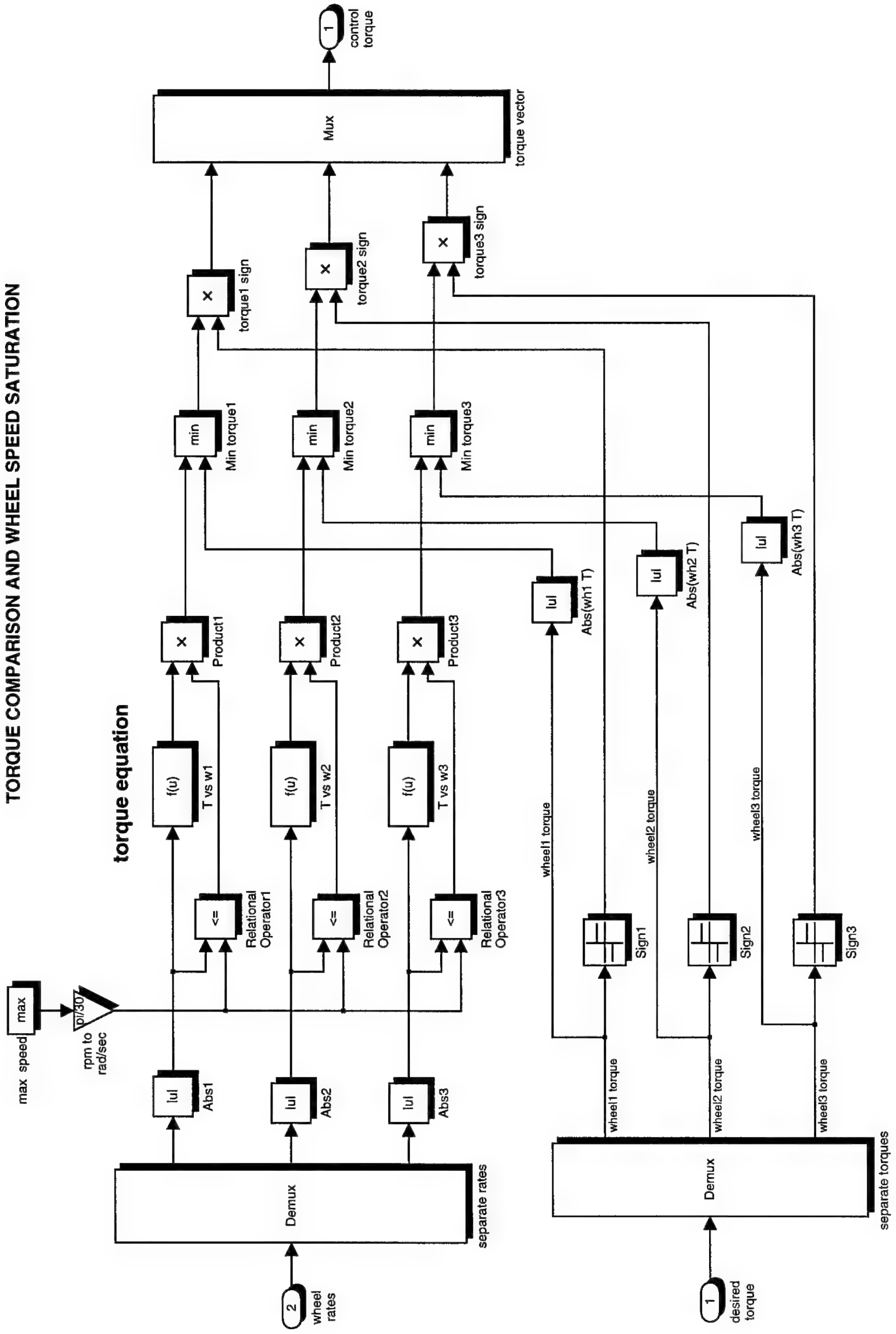


Figure 4.5 SIMULINK Model–Torque Comparison Subsystem

Animatics Corporation, the manufacturer of the motors used in *SIMSAT*, supplied a Torque vs. Motor Speed graph (see Figure 4.6).

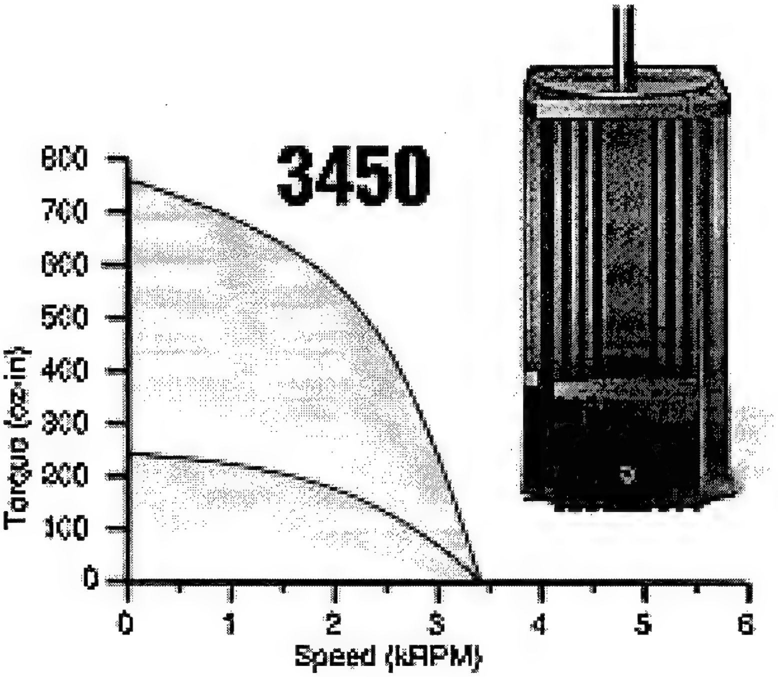


Figure 4.6 Animatics Torque vs. Motor Speed Curves

This graph displays continuous and peak performance when the motors are operating at 48V. Although the peak performance curve is achievable, it greatly stresses the motor and a large portion of power is converted to heat. The torque equation assumes the motors operate along the continuous performance curve. In addition, the motors operate at 36V instead of 48V. The torque curve is scaled down to 36V assuming power input to the motor is approximately equal to power output. Thus, the 36V curve is scaled to 75% of the 48V curve.

Once English units were converted to metric units, the researcher used an exponential curve fit (see Figure 4.7) to approximate the torque equation as

$$T(u) = 0.007061 * (188 - e^{0.0193942u}) \tag{4.1}$$

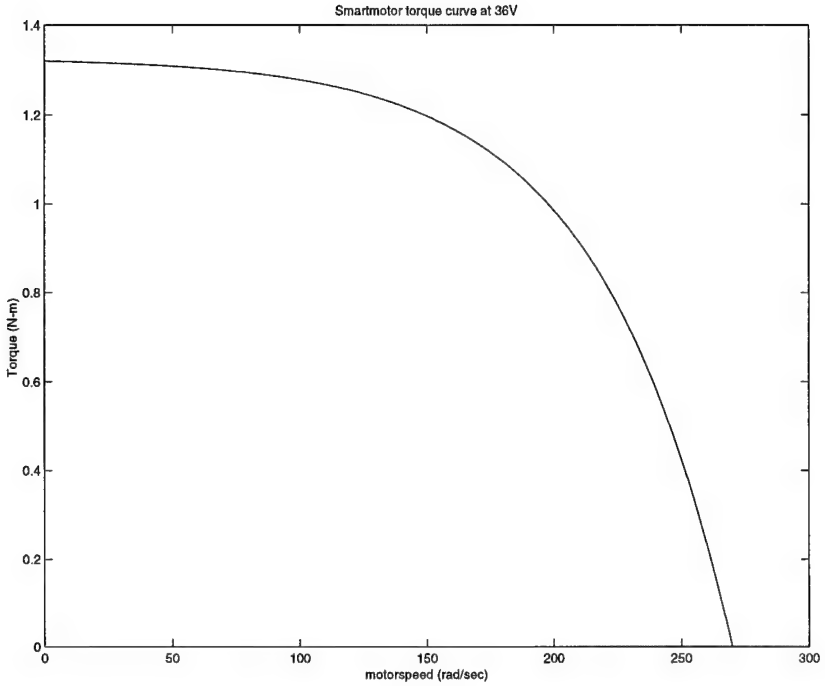


Figure 4.7 36V Torque vs. Motor Speed Curves

4.3.3 ***Static Motor Testing.*** The control torque provided by the controller subsystem is converted into wheel speed. This wheel speed has units of RPM, but the AutoBox accepts inputs in terms of voltage. The author conducted static tests to determine the conversion rates from input voltage to RPMs. The test stand used to support each motor is shown in Figure 4.8.

The cable used to connect the motor to its amplifier while on the test stand differs slightly in configuration from the operational cables. The cable used during the motor testing is 18 feet long while the cables used on *SIMSAT* are only 12 to 18 inches long. Table 4.1 shows the signal cable connection characteristics of the cable used to test the motors.

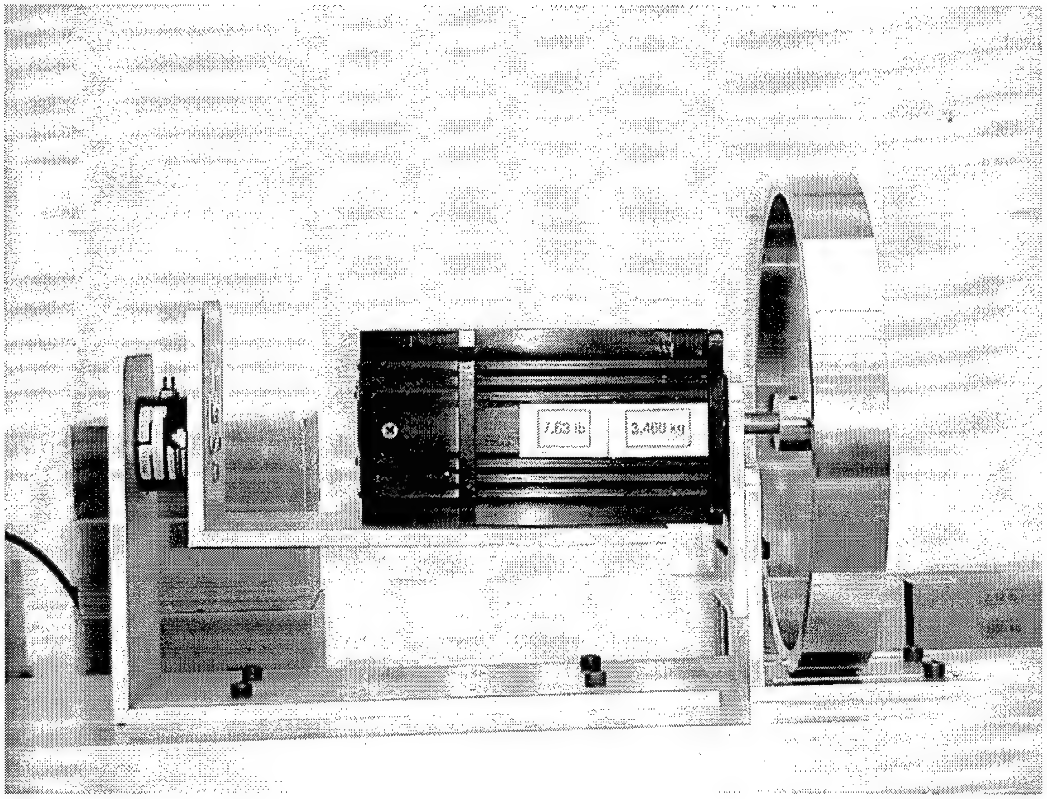


Figure 4.8 Test Stand for Static and Dynamic Motor Testing

Table 4.1 Motor Testing Cable Wiring Placement

Location	Destination	Wire Color
P2 #1	Motor pin 1	Black
P2 #2	Motor pin 2	Red
P2 #3	Motor pin 3	White/Black
P2 #4	Power Bus Bar Ground	Black
P2 #5	Power Bus Bar Voltage	Red
P3 #1	Motor pin 4	Orange
P3 #2	Motor pin 8	Blue/Black
P3 #4	Motor pin 10	Green/Black
P3 #5	Motor pin 5	Blue
P1 #12	Motor pin 12	Orange/Black
P1 #13	Motor pin 13	Blue/White
P1 #14	Motor pin 14	White

The author sent various input voltages to each motor during static testing. Each input voltage resulted in a corresponding wheel speed. The amplifier also generated an output voltage for each change in wheel speed. The author used the root-mean-square method to filter out the noise in the data. Figures 4.9 through 4.14 show the voltage vs. RPM plots for each motor. The slope of each plot is the conversion gains for the SIMULINK model.

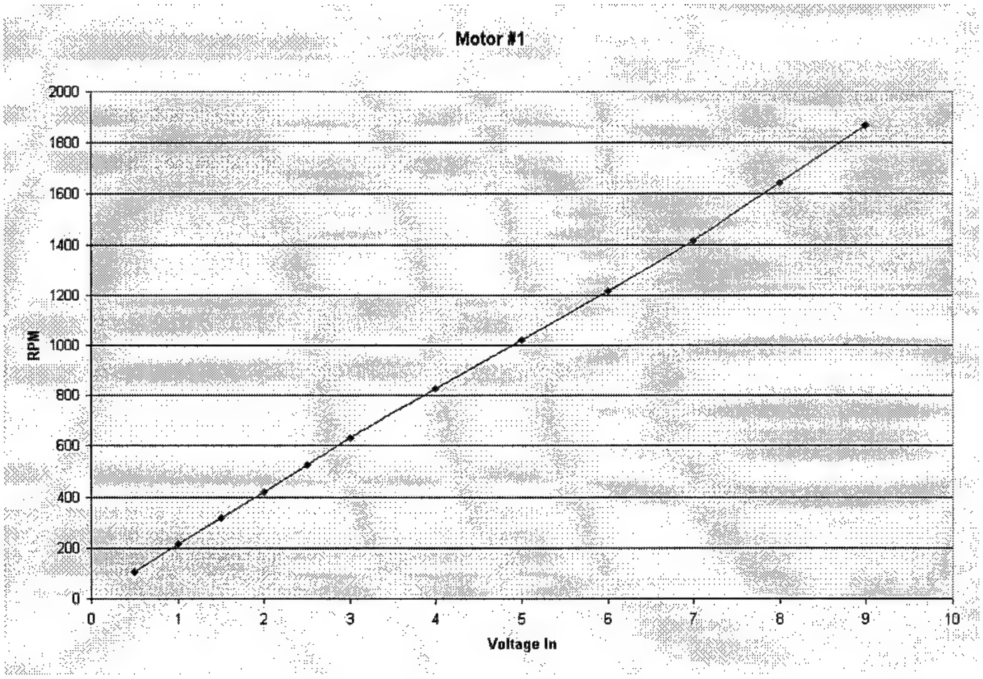


Figure 4.9 Motor #1 RPM vs. Voltage In Plot

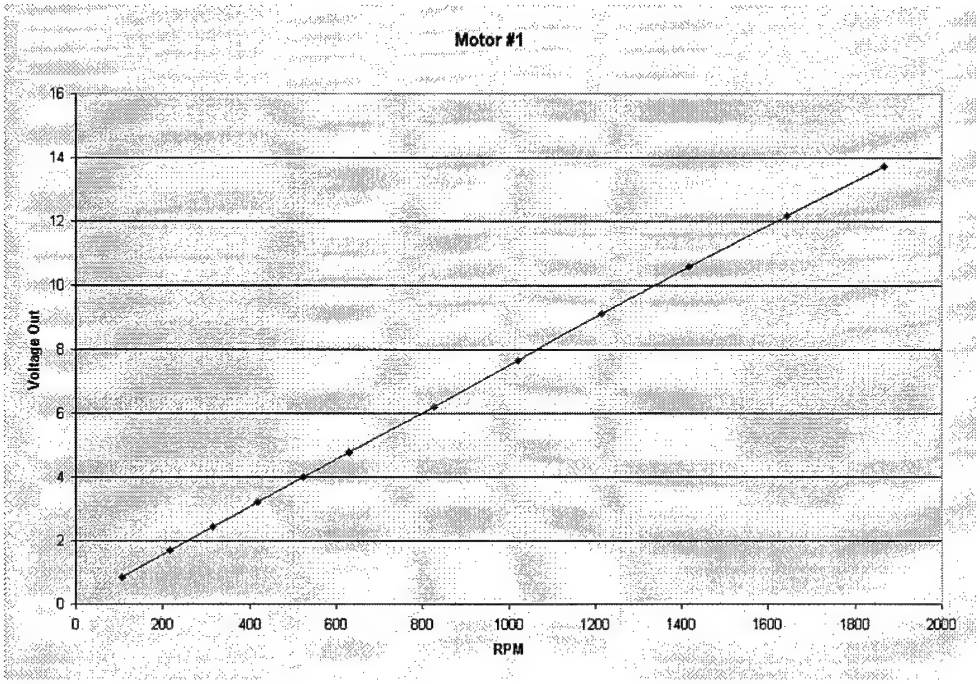


Figure 4.10 Motor #1 Voltage Out vs. RPM Plot

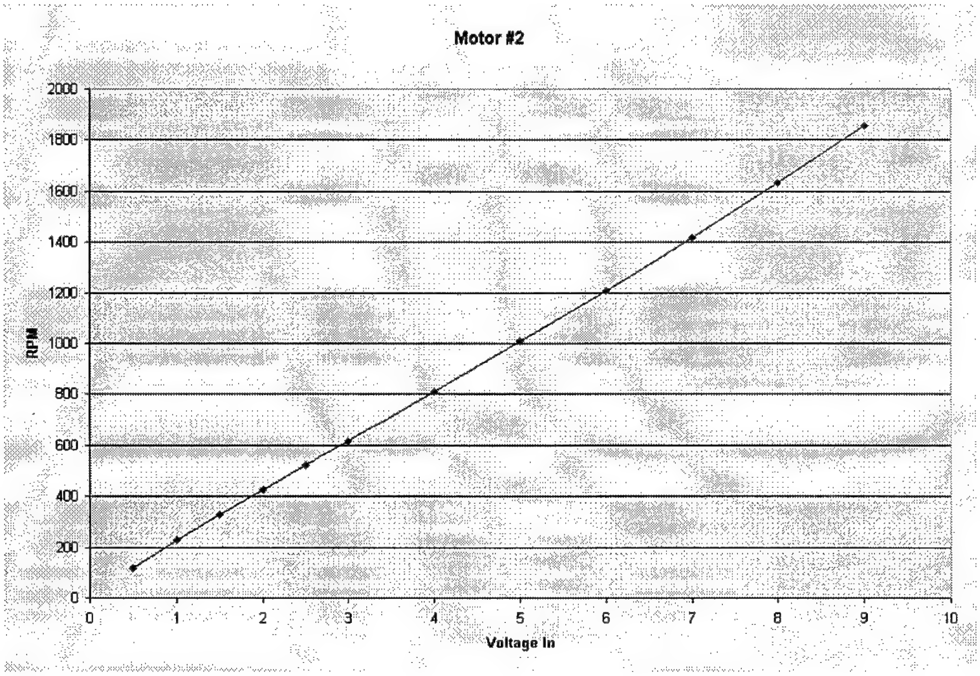


Figure 4.11 Motor #2 RPM vs. Voltage In Plot

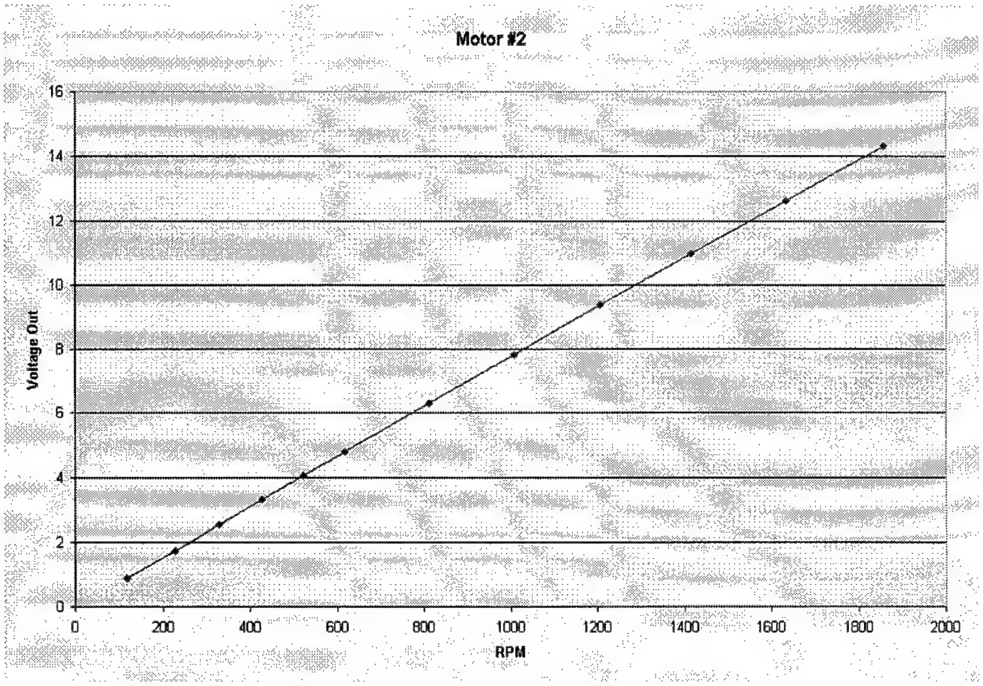


Figure 4.12 Motor #2 Voltage Out vs. RPM Plot

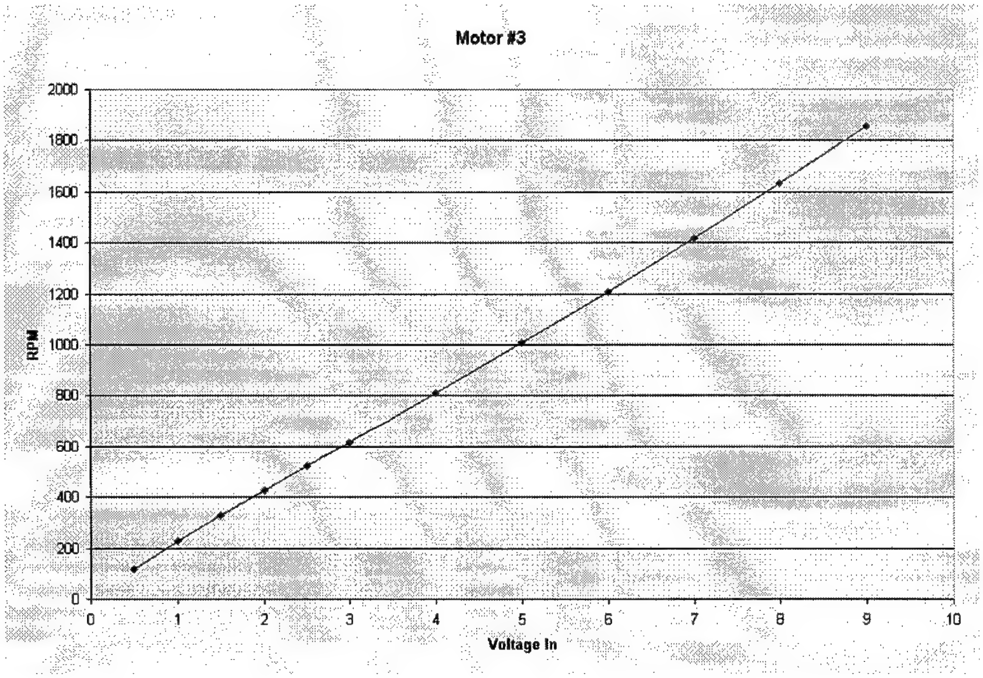


Figure 4.13 Motor #3 RPM vs. Voltage In Plot

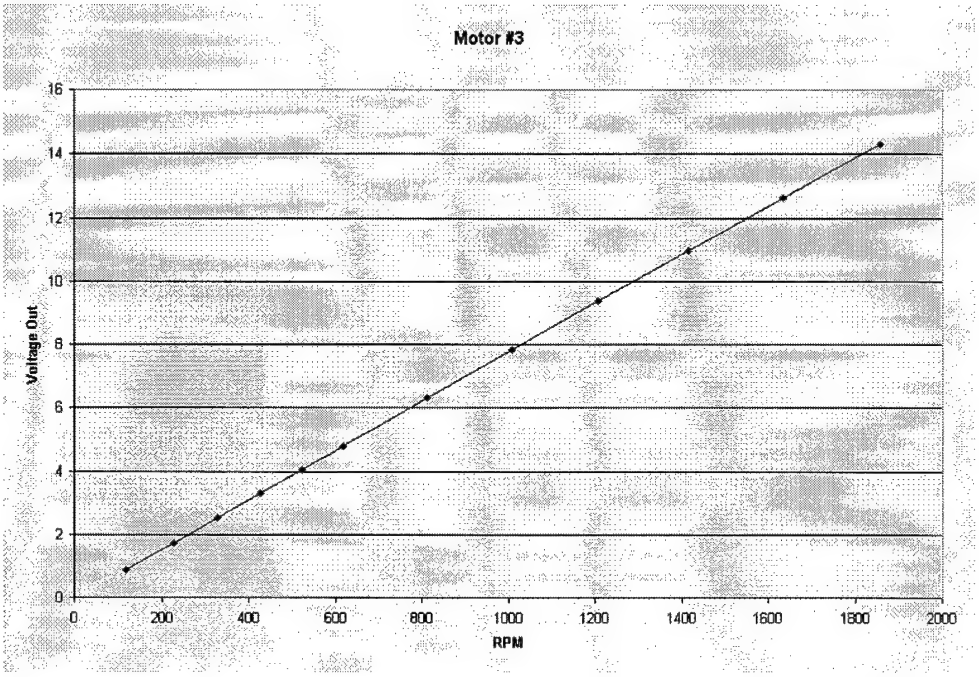


Figure 4.14 Motor #3 Voltage Out vs. RPM Plot

A specific amplifier and momentum wheel is assigned to each motor to provide continuity in system performance. Thus, each amplifier is specifically tuned to its corresponding motor. Turning the loop gain and current limit screws on the amplifier will affect motor response time and maximum RPM. The author was interested in what the response time of the motor was when given a step input. Turning the loop gain and current limit screws clockwise improved response time to within 1.5 seconds (see Figure 4.15). The SIMULINK model, however, assumes an instantaneous response from the motors. Future researchers need to keep this assumption in mind when calculating desired response times for the system.

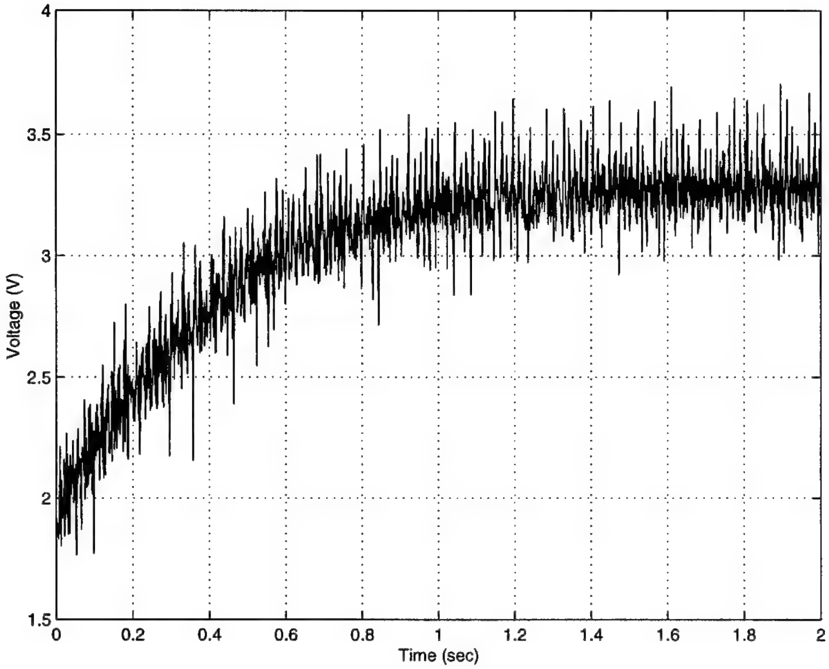


Figure 4.15 Motor #2 Step Response 1 Volt to 2 Volts

Appendix B contains the dynamic motor testing plots of each motor.

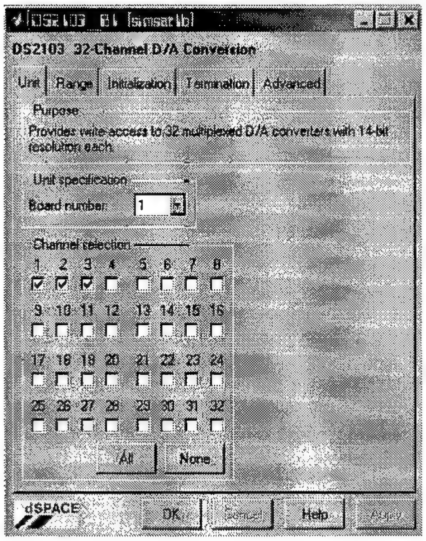
A threshold exists within the loop gain and current limit on the amplifier. Exceeding this threshold may cause the motor to fail to rotate. The motor will generate a high pitch squeal when this occurs. The loop gain and current limit screws are turned counter clockwise to drop below this threshold. It is this threshold which determines the maximum RPM of each motor.

4.3.4 Accessing the AutoBox. The following procedure compiles the SIMULINK model in C-code and downloads it to the AutoBox.

1. On the *SIMSAT* Top Level Software Architecture block diagram select Tools from the top menu
2. Select RTW Options ...
3. Select Solver tab
4. In Solver options Type, select Fixed-step
5. In Solver options, select ode4 (Runge-Kutta)
6. In Solver options Fixed step size, enter 0.001
7. In Solver options Mode, select single tasking
8. Select Real-Time Workshop tab
9. In Build Options Template Makefile, enter rti1003.tmf
10. In Build Options Make Command, enter make_rti
11. Select Build

The AutoBox In/Out section of the top level block diagram is where telemetry data is sent to and received from the AutoBox. Double clicking on the MUX_DAC block will bring up the DS2103 32 Channel D/A Conversion window (see Figure 4.16).

This window allows the researcher to assign output and input signals to various locations on the channel interface board. The three desired momentum wheel rates are inputs for the AutoBox and the outputs received from the AutoBox are the three measured wheel rates and the three angular velocities. Table 4.2 lists the locations where the functional component data lines connect to the channel interface board.



DS2103 32-Channel D/A Conversion

Purpose

Provides write-access to 32 multiplexed D/A converters with 14-bit resolution each.

Unit specification

Board number: 1

Channel selection

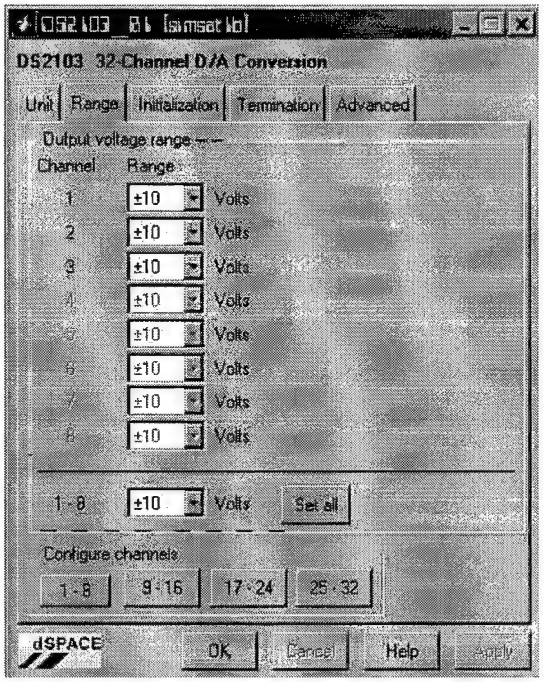
1	2	3	4	5	6	7	8
☑	☑	☑	□	□	□	□	□
9	10	11	12	13	14	15	16
□	□	□	□	□	□	□	□
17	18	19	20	21	22	23	24
□	□	□	□	□	□	□	□
25	26	27	28	29	30	31	32
□	□	□	□	□	□	□	□

Figure 4.16 AutoBox Channel Selection

Table 4.2 Functional Component Data Line Connections to Channel Interface Board

Data Line Number	Input/Output	Position
7 #1	Output	#1
8 #2	Output	#2
9 #3	Output	#3
15 #4	Input	#1
13 #5	Input	#2
14 #1	Input	#3
1 #2	Input	#4
3 #4	Input	#5
5 #5	Input	#6

The DS2103 32 Channel D/A Conversion window is also used to place saturation limits on the momentum wheels by limiting the amount of voltage transmitted through the AutoBox. This voltage range prevents the motors from receiving excessive voltage commands. Figure 4.17 illustrates how a ±10*volts* range is assigned to the telemetry signal.


DS2103_B1 [simsat lib]

DS2103 32-Channel D/A Conversion

Output voltage range

Channel	Range
1	±10 Volts
2	±10 Volts
3	±10 Volts
4	±10 Volts
5	±10 Volts
6	±10 Volts
7	±10 Volts
8	±10 Volts
1 - 8	±10 Volts

Figure 4.17   Limiting AutoBox Telemetry Data

4.3.5 Telemetry Subsystem.

Figure 4.18 represents the telemetry subsystem portion of the SIMULINK model. This block diagram receives telemetry information from the AutoBox and converts the voltage outputs to RPMs for the wheel speeds and degrees per second for angular velocities. The gains used to convert the wheel speeds were generated from the graphs shown in Figures 4.9 through 4.14.

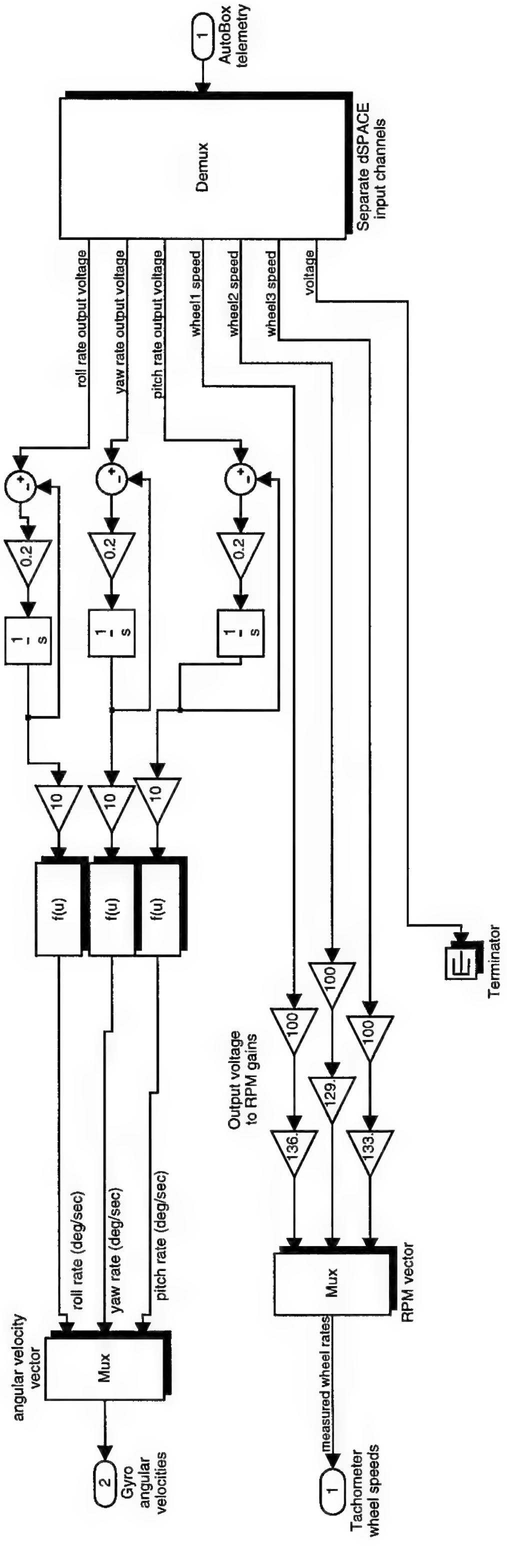
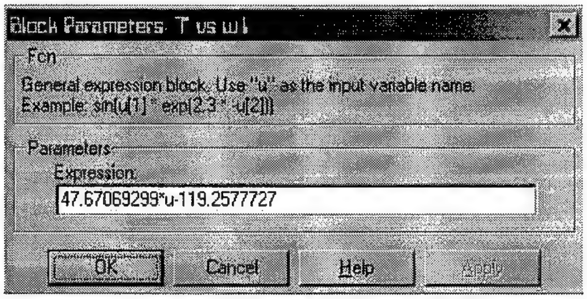


Figure 4.18 Telemetry Subsystem Block Diagram

The manufacturer of the Humphrey CF75 Gyro provided angular velocity response data for *SIMSAT*'s gyro. Excel was used to plot this data and a best-fit curve approximated the equations which convert output voltage to angular rates. Appendix C contains the data and plots. The template used to enter this conversion equation is shown in Figure 4.19.



Block Parameters: T vs ω1

Fcn

General expression block. Use "u" as the input variable name.
Example: sin(u[1] * exp(2.3 * -u[2]))

Parameters

Expression:

47.67069299*u-119.2577727

Figure 4.19 Gyro Rate Conversion Equation Input Window

The gyro outputs between 0 and 5 volts. To determine the zero value for angular rate, the researcher attached an oscilloscope to the output. When the gyro was stationary, the oscilloscope displayed a value of 2.5 volts. Thus, a value between 0 and 2.5 volts represents a negative angular velocity and a positive angular velocity outputs between 2.5 and 5 volts. For some unidentified reason, the AutoBox divides the gyro output by ten. Therefore, the gyro telemetry is multiplied by 10 to obtain true data.

The gyro's output signals experience noise variations of ±4 degrees per second (see Figure 4.20). This noise affects not only the orientation angle calculations but also the wheel speed output and position controller. The author conducted operational tests of *SIMSAT* while the testbed was secured to the support cart. The expected results of these tests should indicate zero angular velocity and no change in inertial orientation of the system.

Figure 4.21 shows the affect the gyro signal noise had on the orientation calculations. The largest deviation occurs in the yaw direction; approximately 0.2 degrees per second.

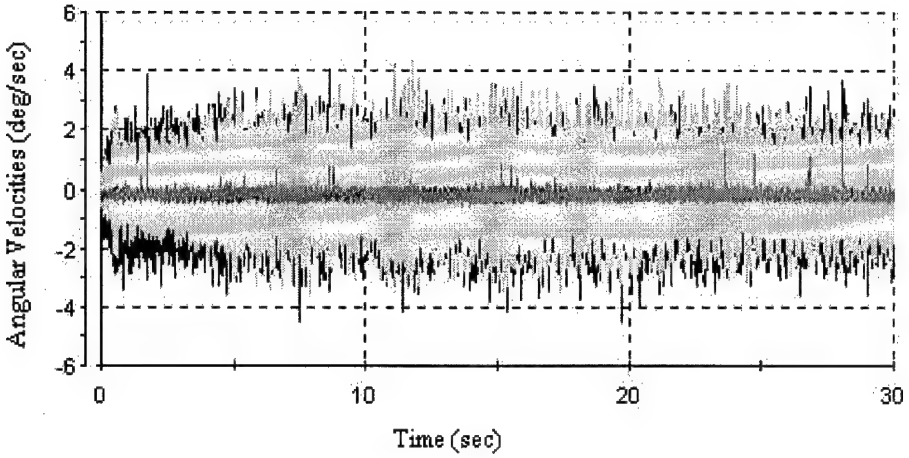


Figure 4.20 Angular Velocity versus Time

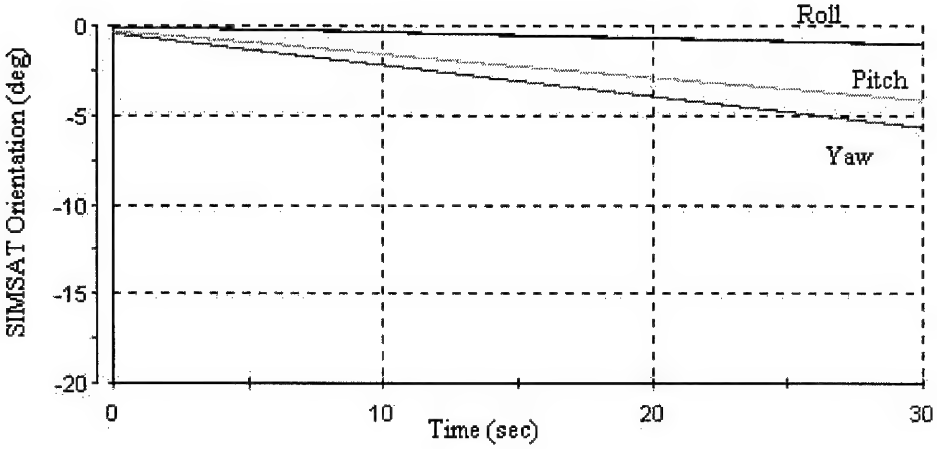


Figure 4.21 SIMSAT Orientation versus Time

A large discrepancy, however, occurs in the commanded wheel speed (see Figure 4.22). The gyro signal noise greatly affects the bandwidth of the wheel speed output. A bandwidth of 850 RPM is measured.

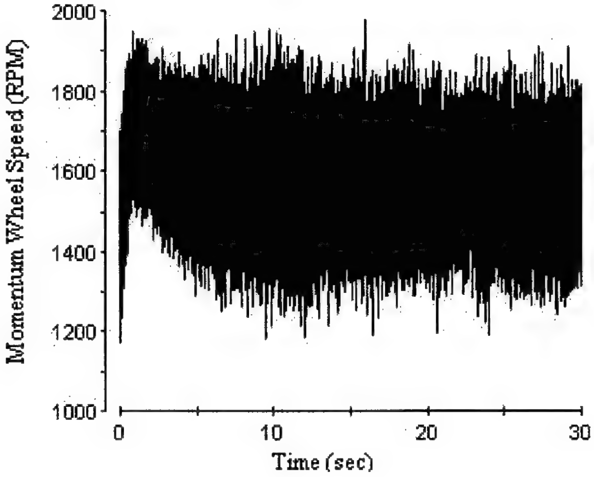


Figure 4.22 Momentum Wheel Speed versus Time

This large bandwidth is unacceptable when wheel speed saturation limits are taken into consideration. This may generate false torque commands in the torque comparison and wheel speed saturation block (see Section 4.3.2).

To reduce the gyro signal noise and thus improve system performance, a low pass noise filter of the form

$$G(s) = \frac{0.2}{s + 0.2} \tag{4.2}$$

was placed on the gyro signal output. Again, the author conducted bench testing of *SIMSAT* to determine the effects the transfer function had on collected data. Uncompensated gyro signal output is shown in Figure 4.23 while the compensated output is shown in Figure 4.24. The transfer function greatly reduced the noise in the gyro signal. However, it took 25 seconds for the signal to reach a steady-state value.

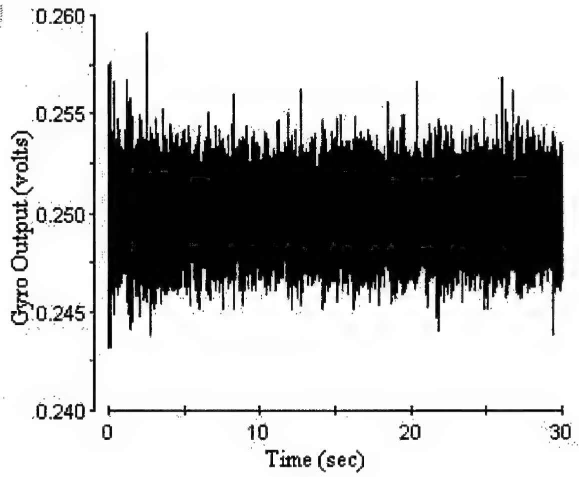


Figure 4.23 Uncompensated Gyro Output Telemetry

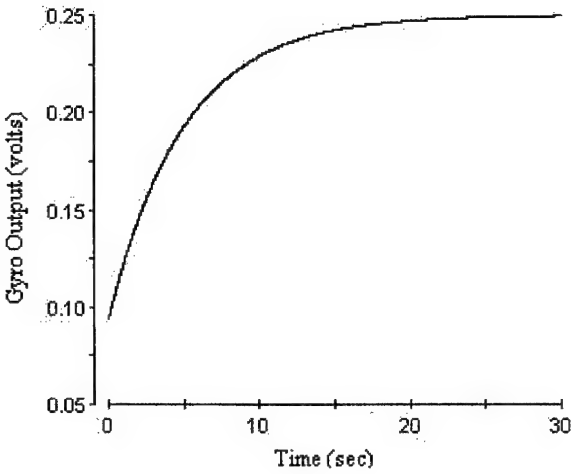


Figure 4.24 Compensated Gyro Output Telemetry

The noise bandwidth in wheel speed improved 340% as a result of adding the low pass noise filter to the telemetry output (see Figure 4.25).

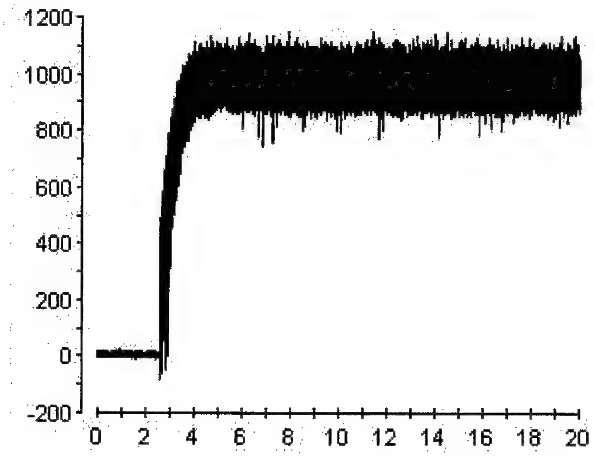


Figure 4.25 Compensated Momentum Wheel Speed versus Time

According to the above figure, the SIMULINK model was setting the initial gyro data value as zero. This initial value had adverse affects on system response. Figure 4.26 illustrates how the zero initial condition altered angular velocity readings.

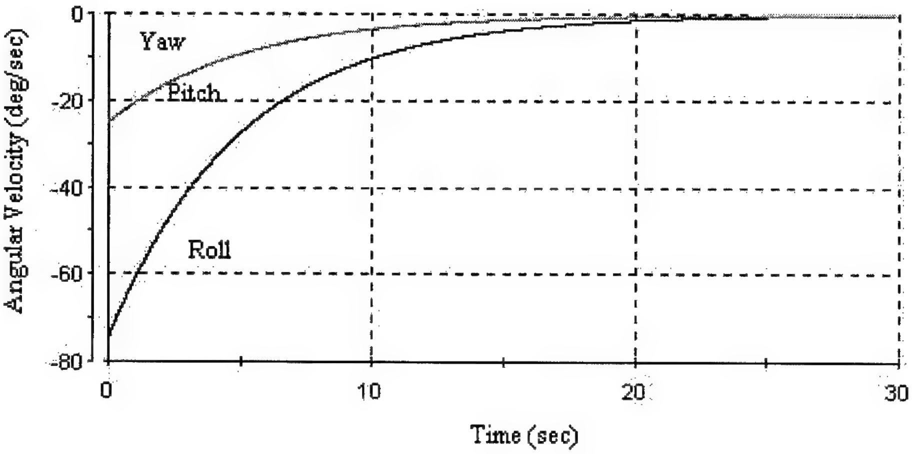


Figure 4.26 Angular Velocity versus Time

Note how none of the angular velocities begin at zero. These non-zero values produced the inertial orientation plots shown in Figure 4.27.

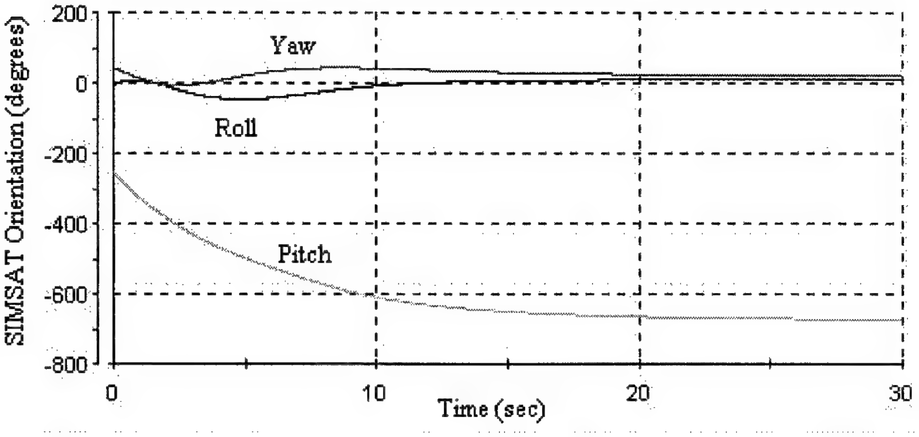


Figure 4.27 *SIMSAT* Orientation versus Time

The zero initial condition for the noise filter generated a pitch maneuver of 670 degrees. This is inaccurate since the system is restricted from moving. Therefore, an initial condition of zero for the gyro output signals is an incorrect assumption. As stated earlier in this section, an output of 2.5 volts represents a zero angular velocity value. The above transfer function only accepts an initial condition of zero. Thus, the author used an alternate method to build the low pass noise filter.

The new method of building the lag compensator uses an integration loop as shown in Figure 4.28.

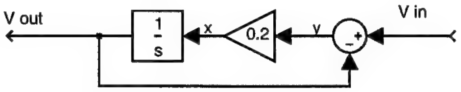


Figure 4.28 Integration Loop for Lag Compensation

The mathematical representation of the block diagram is as follows:

$$\begin{aligned} y &= Vin - Vout \\ x &= 0.2y \\ Vout &= \frac{1}{s}x \end{aligned}$$

$$\begin{aligned}\dot{V}out &= x \\ &= 0.2y \\ &= 0.2(Vin - Vout) \\ \dot{V}out + 0.2Vout &= 0.2Vin \\ \frac{Vout(s)}{Vin(s)} &= \frac{0.2}{s+0.2}\end{aligned}$$

The integration loop provides the identical transfer function as used previously. The integration block, however, includes the ability to set the initial conditions to something other than a zero value. The addition of this integration loop alters the previous angular velocity plot (see Figure 4.29).

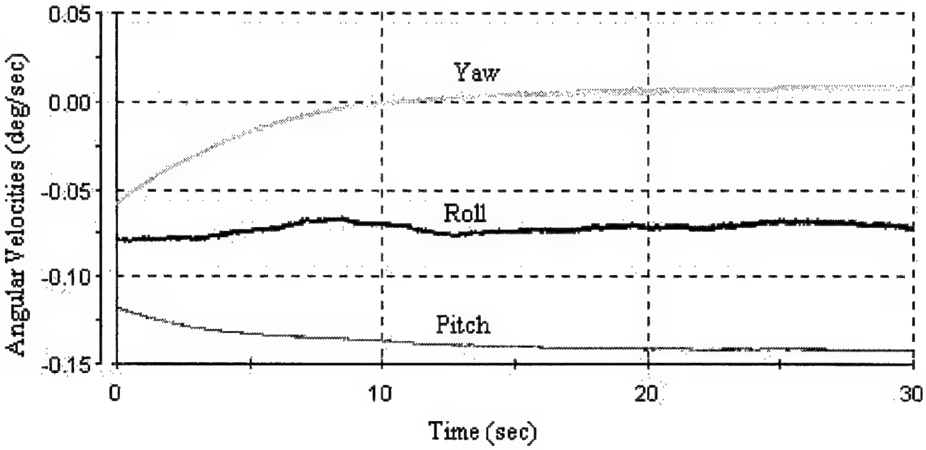


Figure 4.29 Angular Velocities after Integration Loop

Now, all angular velocities reach a steady-state value of approximately zero in half the time (12 secs) than they did while using the transfer function. This improvement in system performance allows for a better estimate of inertial orientation (see Figure 4.30).

The average improvement in estimating inertial orientation is 133% in accuracy. The drift in orientation calculations has also improved by 125%.

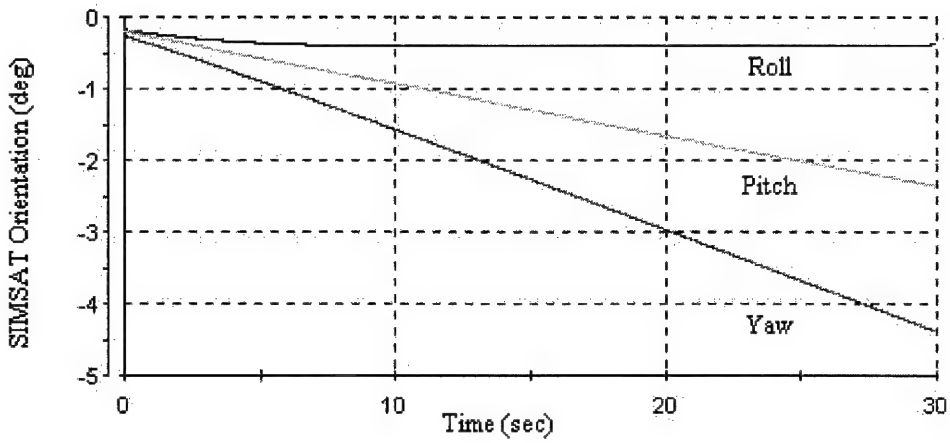


Figure 4.30 *SIMSAT* Orientation after Integration Loop

4.3.6 Euler Transformation Block.

The final block of interest in the SIMULINK model is the Euler transformation block (see Figure 4.31). This block represents the (1-3-2) rotation sequence, Eq. 2.17, in block diagram format. The measured angular velocities combine with the inertial orientation angles calculated in a previous iteration to generate current roll, yaw, and pitch velocities. These angular velocities are integrated to calculate the current inertial orientation of *SIMSAT*. These orientation angles are sent back to the Controller Subsystem block to continue with the control process.

4.4 Graphical User Interface

The SIMULINK model processes the various control laws and calculations used to operate *SIMSAT*, but Control Desk allows the researcher to manually input and receive telemetry data. Figure 4.32 is the Graphical User Interface (GUI) which the researcher uses to operate *SIMSAT*.

The researcher can input desired roll, yaw, and pitch commands by using slider bars, incremental inputs, or direct numerical inputs. Digital displays show the current orientation as well as commanded orientation. A push button is provided to simplify returning input commands to zero. Gauges allow the researcher to visualize measured angular velocities and momentum wheel speeds. Digital displays located directly below the gauges display exact measurements of these values. The researcher may also use Control Desk to graph telemetry data (see Figure 4.33). This capability also stores telemetry data to a file for future analysis.

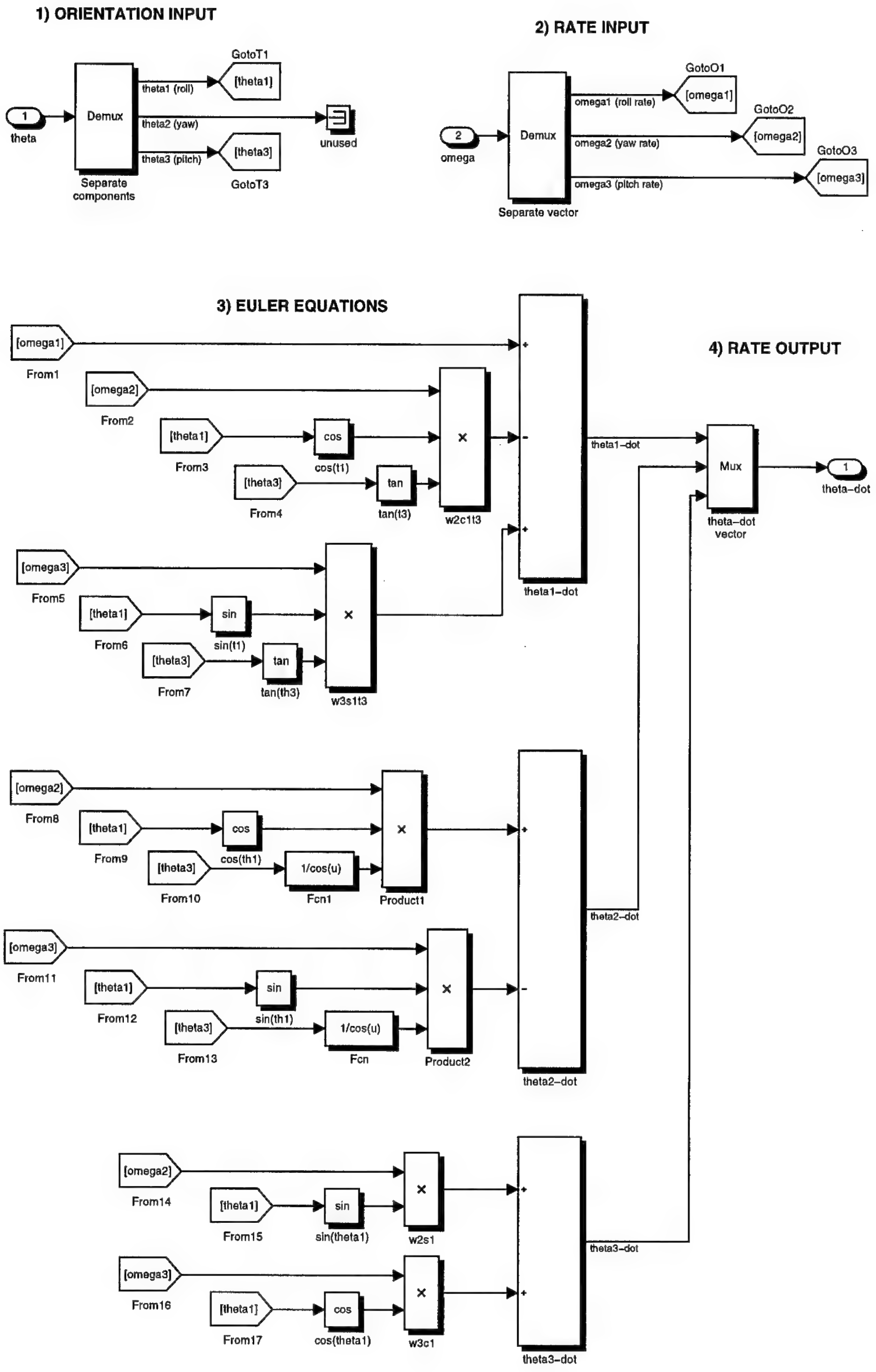
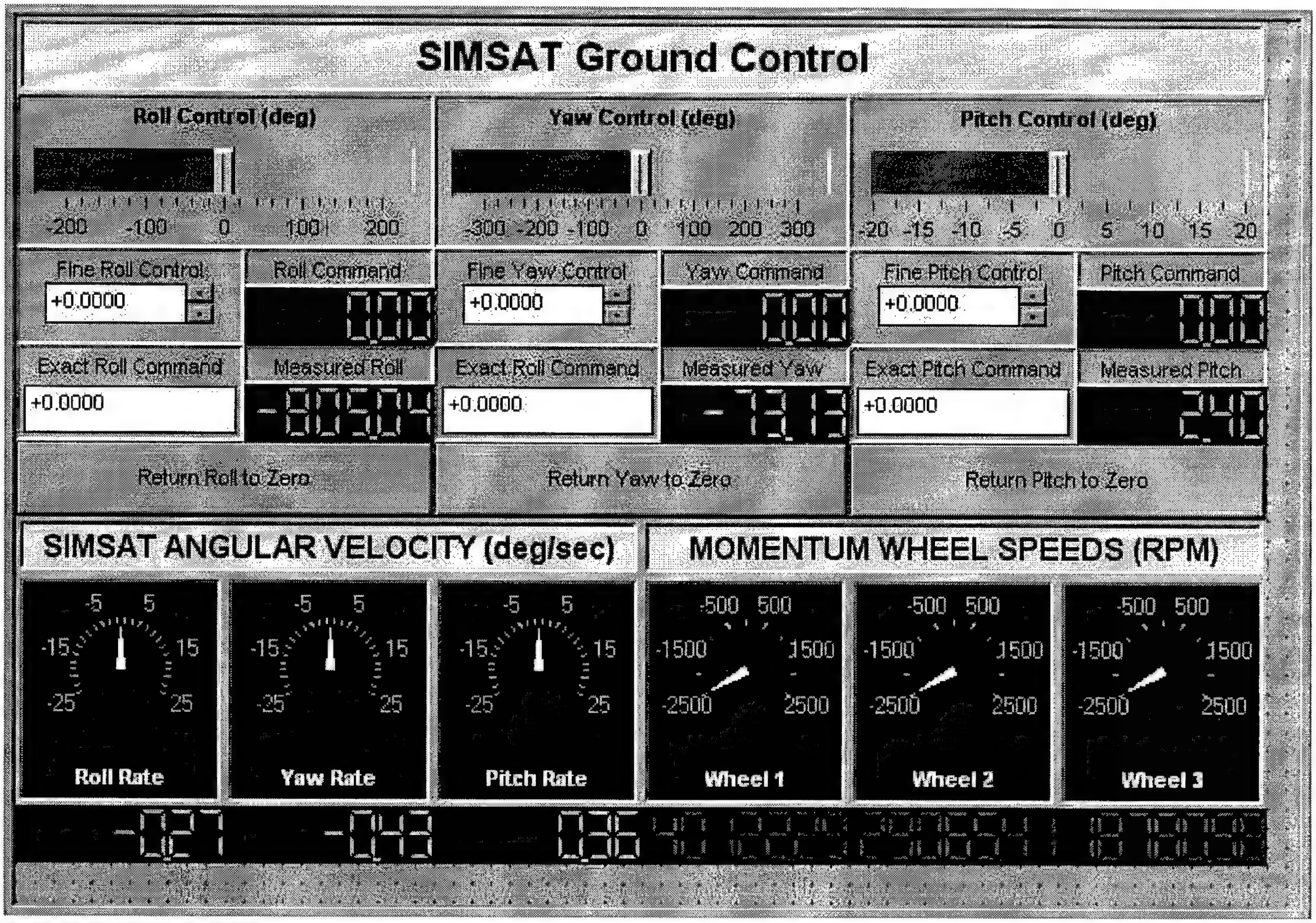


Figure 4.31 Euler Transformation Block Diagram



SIMSAT Ground Control

Fine Roll Control: +0.0000

Roll Command: 0.00

Exact Roll Command: +0.0000

Measured Roll: -80.504

Fine Yaw Control: +0.0000

Yaw Command: 0.00

Exact Roll Command: +0.0000

Measured Yaw: -73.13

Fine Pitch Control: +0.0000

Pitch Command: 0.00

Exact Pitch Command: +0.0000

Measured Pitch: 2.40

Roll Rate: -0.27

Yaw Rate: -0.43

Pitch Rate: 0.36

Figure 4.32 Control Desk Operating Panel

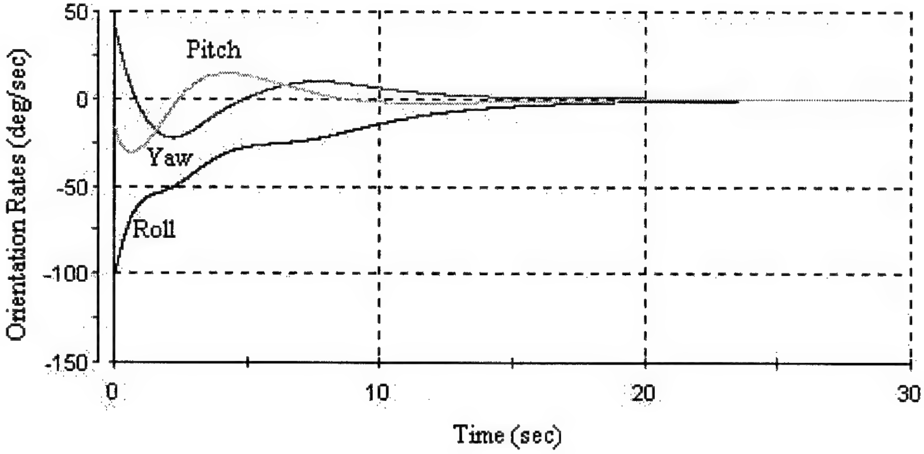


Figure 4.33 Plot of Incoming Telemetry Data

The following procedure is used to open Control Desk's GUI.

1. Begin Control Desk
2. On the menu bar select File
3. Select Open Experiment
4. The file is located at C:/Simsat Model/Simsat Model.cdx
5. Select OK

The researcher must ensure the AutoBox is operating prior to opening Control Desk. If the AutoBox is not operating, Control Desk will change the hardware connection between itself and the AutoBox. To reset the hardware connection, allowing the RadioLAN to transmit and receive data, accomplish the following:

1. Select the Hardware tab at the bottom of the Simsat Model window
2. Select Hardware from the main menu
3. Select Change Connection ...
4. In the Choose Connection block select Net
5. Ensure the Network Client is 192.100.100.98
6. Select OK

After completing this procedure, Control Desk will attempt to reconnect to the AutoBox using the RadioLAN. If the connection fails an error message is printed in the log viewer window. This message reads "Encountered a network error. Please check if the net box is switched on". Attempt the procedure one more time. If the message appears again then ping the AutoBox using the procedure from Section 4.2. If the RadioLAN does not respond, it is possible the DockLINK is no longer functioning.

If the researcher wishes to use the EtherLink connection, then in the Choose Connection block select Bus. The EtherLink is the only method used to connect to the AutoBox when the DockLINK fails to operate.

4.5 Multimedia Lesson Plan

The author developed a multimedia lesson plan to assist AFIT instructors in teaching basic satellite dynamics concepts. Air Force Manual 36-2236 contains information on preparing the lesson plan [32]. The lesson plan combines computer-based presentation with video of *SIMSAT* in motion. The author utilized Microsoft PowerPoint to develop the multimedia lesson slides (see Appendix D). The objective of the lesson and the lesson outline are provided to aid the instructor in presenting the lecture.

In addition to the multimedia lesson plan, the author also generated instructions on conducting a lab experiment. This lab experiment offers students the opportunity to gain hands-on experience in maneuvering *SIMSAT*. The experiment reinforces the information presented in the lesson plan. A copy of the experiment is located in Appendix D.

4.6 Estimating Inertia Properties

The 1999 Graduate Systems Engineering Team used inerbal.m (see Appendix A) to calculate the inertia matrix for *SIMSAT*. The principal inertia values they came up with were

$$\begin{aligned} I_{roll} &= 4.488 kgm^2 \\ I_{pitch} &= 15.971 kgm^2 \\ I_{yaw} &= 17.361 kgm^2 \end{aligned}$$

These values were not obtained through simulation but by using AutoCAD drawings. To better calculate the inertia properties of *SIMSAT*, the author conducted open loop experiments of the system. In open loop response analysis, the author commanded a single momentum wheel and measured the angular velocity of the system about the commanded axis. Various wheel speed were commanded in both a positive and negative direction. Control Desk displayed the corresponding angular velocity to each momentum wheel rate (see Tables 4.3 through 4.5).

Table 4.3 Roll Axis Angular Velocity Response

Momentum Wheel Speed (RPM)	Angular Velocity (deg/sec)
106	-2.444
216	-3.848
-106	2.742
-216	3.319

Table 4.4 Pitch Axis Angular Velocity Response

Momentum Wheel Speed (RPM)	Angular Velocity (deg/sec)
105	-0.492
217	-0.912
-105	0.496
-217	0.900

Table 4.5 Yaw Axis Angular Velocity Response

Momentum Wheel Speed (RPM)	Angular Velocity (deg/sec)
59	-0.247
117	-0.488
-59	0.229
-117	0.490

The principal inertia values for *SIMSAT* were calculated using the following equation:

$$I = -\frac{J(\dot{\theta} + \Omega)}{\dot{\theta}} \tag{4.3}$$

where

$$\begin{aligned} I &= \text{inertia of satellite} \\ J &= \text{inertia of wheel} \\ \Omega &= \text{angular rate of wheel relative to the satellite} \\ \dot{\theta} &= \text{angular rate of satellite} \end{aligned}$$

and averaged the inertia values for each test case. The calculated principal inertia properties of *SIMSAT* are:

$$\begin{aligned} I_{roll} &= 5.914kgm^2 \\ I_{pitch} &= 26.421kgm^2 \\ I_{yaw} &= 29.084kgm^2 \end{aligned}$$

These inertia values differ from the estimated values because several modifications were made to the overall system design.

4.7 Summary

The author developed several procedures during the experimental testing of *SIMSAT*. Some of these procedures dealt with connecting the ground station PC to the AutoBox using either the RadioLAN or the EtherLink connection. Other procedures provided steps for establishing a connection between the AutoBox and the software applications used to operate *SIMSAT*. These software applications contain the dynamic and stabilization control equations for *SIMSAT* as well as the graphical user interface needed to transmit and receive telemetry data. A multimedia lesson plan and lab experiment utilizes the operational system in demonstrating satellite dynamics concepts to the students of AFIT.

During the experimental testing of the SIMULINK model, the following assumptions were made:

1. The 1.5 second delay in the motors reaching steady-state has little impact on system calculations
2. No external torques are acting on *SIMSAT*
3. The system is unaffected by the angular velocity of the Earth
4. Friction losses from gyro bearings and the air bearing assemble are negligible
5. The change in angular velocity versus change in momentum wheel speed is linear

V. Research Summary and Recommendations

5.1 Research Summary

The purpose of this research effort was to construct a realistic space-platform simulator for experimentation and a multimedia educational tool. The construction phase took approximately 300 hours to complete. During this phase, structural, functional, and software components were successfully integrated to create an operational laboratory-based satellite simulator.

5.2 System Design Modifications

Although *SIMSAT* is operational, the author suggests that some modifications to the system's design be made to improve experimental results. The gyro is currently mounted directly to a mounting plate. This may allow low level vibrational noise to interfere with the gyro's output signals. A low pass noise filter was designed into the SIMULINK control model to reduce the high frequency noise but it is ineffective against low level vibrations. It may prove beneficial to mount the gyro on top of rubber mounts. This may aid in protecting the gyro's sensitive hardware. Another suggestion is to construct an estimator, such as a Kalman filter, to replace the low pass noise filter.

A systematic characterization of the functional components may also prove beneficial in determining overall system operational characteristics. Removing the gyro from *SIMSAT* and conducting bench tests on output signals will aid future researchers in determining the operational characteristics of the gyro and how noise is induced into the output signal. The batteries can also be tested to determine how 'clean' of a power source is available to the functional components of *SIMSAT*.

The author also experienced some complications while using the wireless RadioLAN. After using the system for a month, the DockLink mounted on *SIMSAT* ceased to transmit and receive telemetry data. So far, technical support has been unable to provide any information on what may cause the system to fail. Some suggestions provided by Mr. Jay Anderson are:

- Reposition the 36V wire which currently rests on top of the DockLink
- Regulate the incoming voltage to prevent voltage spikes from affecting the DockLink's performance

According to Charles Elachi, "radio frequency waves are usually generated by periodic currents of electric charges in wires," [9]. A time varying electric field will also generate a magnetic field. This magnetic field, or the radio waves, may also cause the system to fail. It is suggested that future researchers relocate the DockLink away from the batteries.

Another system design modification is to drill holes into the Lexan box protecting the motors and momentum wheels. As heat builds up within the box, the response times of the motors may increase. This increase in temperature may also affect the optimal torque output of the momentum wheels.

Momentum wheels are not the only method of controlling a three-axis stabilized spacecraft. Recall in Section 2.4.4 that control moment gyros and thrusters are also capable of meeting control requirements. Adding these items to *SIMSAT* will offer increased opportunities for future research.

5.3 Future Research Recommendations

This research effort has not come close to utilizing the full capabilities which *SIMSAT* offers. The experimental and educational potential of this system far exceeds what is covered within this document. Continued use of *SIMSAT* will aid research conducted at AFIT in the following areas:

- Satellite pointing and tracking
- Station keeping experimentation
- Rigid and flexible structure experimentation
- Power source research and development

- Development of improved spacecraft controllers
- Thruster, momentum wheel, and control moment gyro research and development
- Wireless communications research
- Additional multimedia educational products
- Interactive educational internet package

5.4 Conclusion

This research effort, and future research efforts utilizing *SIMSAT*'s capabilities, will strengthen AFIT's space operations and astronautical engineering curriculums. But other departments, such as the physics department, may also find *SIMSAT* a useful research tool. Space is a valuable resource which the USAF needs to effectively access. Only by fully understanding this resource will today's air force become the premier air and space force by the year 2025.

Appendix A. MATLAB Files

The MATLAB files related to the equations of motion used to model *SIMSAT* are contained in this appendix for future researchers to reference. The following files were created by the 1999 Graduate Systems Engineering Team [7]:

- **"inerbal.m"**
- **"quikiner.m"**
- **"simcloop3.m"**
- **"domega2.m"**
- **"solver2play.m"**
- **"domega.m"**

These files are located on the ground station PC in the C:/simsat file folder.

A.1 inerbal.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% "inerbal.m" utilizes the functions of AutoCAD to develop the composite SIMSAT        %
% inertia matrix with respect to SIMSAT's center of mass.  This program                %
% can be paired with SIMSATFINAL1nopayldpltcwmechBALANCE.dwg and                        %
% SIMSATFINAL1withobjectgrouping&8RODS.dwg (or other yet-to-be-created SIMSAT          %
% drawings) in AutoCAD.  "inerbal.m" also calculates the composite center of mass of   %
% SIMSAT and aids the user with balancing SIMSAT (i.e., attempting to make the         %
% position vector FROM the origin at the center of the sphere TO the composite center  %
% of mass [0;0;0]).  "inerbal.m"  will prompt the user for the horizontal distance     %
% between the base plate center of mass (c.o.m.) and the c.o.m. of a mounting plate in %
% order to determine the position vectors of the various components (gyro, motors,     %
% batteries, amplifiers, etc., etc.). This program also determines the inertia of each %
% momentum wheel (using the 8 & 5/8" steel hoop and 1/4" aluminum disk baseline design)%
% because the wheel inertias are required for the SIMULINK and MATLAB motion           %
% simulations.  "inerbal.m" concludes by calculating the inverse matrix of the first   %
% four terms (these terms do not vary with time) of the equations of motion (eom).     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                                   %
% This code was developed to create inertia matrices and transformation matrices    %
% for Simsat components which are involved within the equations of motion.  The     %
% components of the SIMSAT system include:                                          %
%                                                                                   %
% 1)  Hollow Central Sphere                                                         %
% 2)  One Hollow Central Arm                                                        %
% 3)  Two Mounting Collars                                                          %
% 4)  Two Base Plates                                                               %
% 5)  Six Mounting Plates                                                           %
% 6)  Eight Support Rods                                                            %
% 7)  Autobox (and mounting equipment)                                              %
% 8)  Battery(ies) and housings                                                     %
% 9)  Transmitter/Receiver                                                          %
% 10) Antenna                                                                       %
% 11) Gyro and housing                                                              %
% 12) Three Momentum Wheels                                                         %
% 13) Three Momentum Wheel Motors                                                   %
% 14) Three Amps                                                                    %
% 15) Lexan Box                                                                     %
% 16) MW Shelves                                                                    %
% 17) Mythical Counterweight                                                        %
%                                                                                   %
% Reference Frames:                                                                 %
% 1) Inertial Reference Frame--x axis points to the north laboratory wall (or other %
%    convenient wall), y axis points at lab ceiling, z axis is deduced from         %
%    right-handed orthogonality.                                                    %
% 2) SIMSAT Body-Fixed Frame (the "b" basis set with origin at center of sphere)    %
% 3) Each momentum wheel has its own body-fixed reference frame (d, f, and h bases) %
% 4) The steel counterweight blocks have their own body-fixed reference frame       %
%    (u bases)                                                                      %
%                                                                                   %
% Bases:                                                                            %
% 1) "b basis" (body-fixed basis, origin is at center of central sphere)            %
%    b1 points to the right of this screen                                          %
%    b2 points to the top of this screen                                            %
%    b3 points out of the screen                                                    %
% 2) "d, f, and h bases"--centered on momentum wheels 1, 2 and 3, respectively,     %
%    centers of mass (C.O.M.). If momentum wheel is lying flat on a table, d2, f2   %
%    and h2 point up (parallel with wheel axle hole). d1/f1/h1 and d3/f3/h3 are     %
%    right-handed orthogonal with d2/f2/h2.  The exact pointing directions of       %
%    d1/f1/h1 and d3/f3/h3 are not important as long as right-handed orthogonality  %
%    with d2/f2/h2 is satisfied.                                                    %
% 3) "u bases" (centered on a counterweight block's C.O.M.)                         %
%                                                                                   %
% Assumptions                                                                       %
% 1) Each component has uniform density.                                            %
% 2) Steel counterweight blocks are modeled as rectangular parallelpipeds           %
% 3) Mass and inertia of electrical wires and clamp-on collars NOT included         %
% 4) After balancing, SIMSAT's center of mass is colocated with the center          %
%    of the sphere ([0;0;0] in the "b" reference frame)                             %
%                                                                                   %
% Sections within the code                                                          %
%                                                                                   %
% Section 1 -- Mass and position vectors for the different components.              %
% Section 2 -- Counterweight is positioned using center of mass calculations        %
%              and, if necessary, the inertia matrix for the counterweight block(s) %
%              is calculated and adjusted using the parallel axis theorem.          %
% Section 3 -- Calculates inertia matrices of the momentum wheels                   %
% Section 4 -- Inertia matrix from Autocad input by the user.                       %
% Section 5 -- Calculates SIMSAT total mass, SIMSAT composite inertia matrix, and   %
%              inverse matrix of the first four terms (these terms do not vary with %
%              time) of the equations of motion (eom).                              %
%                                                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                                   %
% Definitions of Terms                                                              %
%                                                                                   %
% Inertia Matrices (in kg*m^2):                                                     %
%  Icomp - composite SIMSAT inertia matrix about the SIMSAT center of mass          %
%                                                                                   %
%  Jwheel1 - Inertia of wheel 1 about the wheel 1 center of mass w.r.t. d basis     %
%  Jwheel2 - Inertia of wheel 2 about the wheel 2 center of mass w.r.t. f basis     %
%  Jwheel3 - Inertia of wheel 3 about the wheel 3 center of mass w.r.t. h basis     %
%                                                                                   %
%  Jw1w1b  - Inertia of wheel 1 about the wheel 1 center of mass w.r.t. b basis     %
%  Jw2w2b  - Inertia of wheel 2 about the wheel 2 center of mass w.r.t. b basis     %
%  Jw3w3b  - Inertia of wheel 3 about the wheel 3 center of mass w.r.t. b basis     %
%                                                                                   %
%  Jw1cb - Wheel 1 inertia matrix about the SIMSAT center of mass wrt b basis       %
%  Jw2cb - Wheel 2 inertia matrix about the SIMSAT center of mass wrt b basis       %
%  Jw3cb - Wheel 3 inertia matrix about the SIMSAT center of mass wrt b basis      %
%                                                                                   %
% Transformation Matrices:                                                          %
%  rotaw1 - Transformation Matrix for moving from the d basis to the b basis        %
%  rotaw2 - Transformation Matrix for moving from the f basis to the b basis        %
%  rotaw3 - Transformation Matrix for moving from the h basis to the b basis        %
%  rotbw1 - Transformation Matrix for moving from the b basis to the d basis        %
%  rotbw2 - Transformation Matrix for moving from the b basis to the f basis        %
%  rotbw3 - Transformation Matrix for moving from the b basis to the h basis        %
%                                                                                   %
% Some Important Position Vectors:                                                  %
%  rcomp   - Position vector (in meters, written in the "b" basis) FROM the origin  %
%              at center of central sphere TO the SIMSAT composite center of mass   %
%  rmw1v   - Position vector FROM center of sphere TO center of mass of momentum    %
%              wheel #1 in meters (written in the "b" frame)                        %
%  rmw2v   - Position vector FROM center of sphere TO center of mass of momentum    %
%              wheel #2 in meters (written in the "b" frame)                        %
%  rmw3v   - Position vector FROM center of sphere TO center of mass of momentum    %
%              wheel #3 in meters (written in the "b" frame)                        %
%                                                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Section 1 -- Mass and position vectors for the different components %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Masses of each component, ALL MASSES GIVEN IN KILOGRAMS:                %
%                                                                         %
% StructurAL Components                                                   %
%   mcsca      - Mass of central sphere and central arm                   %
%   mmc(1-2)   - Mass of mounting collar                                  %
%   msr(1-8)   - Mass of support rod 1 through 8                          %
%   mbp(1-2)   - Mass of 1/2 inch base plate 1 through 2                  %
%   mmp1       - Mass of the momentum wheel mounting plate                %
%   mmp2       - Mass of the Lexan box mounting plate                     %
%   mmp3       - Mass of the two-battery mounting plate                   %
%   mmp4       - Mass of the gyro/battery/transceiver mounting plate      %
%   mmp5       - Mass of the AutoBox mounting plate                       %
%   mpp        - Mass of the payload (pegboard) mounting plate            %
%                Note: The payload plate may not always be used           %
%   msc        - Total mass of all structural components                    %
%                                                                          %
% Functional Components                                                    %
%   mA         - Mass of the AutoBox                                        %
%   mub        - Mass of the AutoBox's U-bracket                            %
%   mlb        - Combined mass of the 4 AutoBox L-brackets                  %
%   mtr        - Mass of transmitter/receiver                               %
%   mant       - Mass of antenna                                            %
%   mbt(1-4)   - Mass of battery 1 through 4                                %
%   mbt3hs     - Mass of battery #3 support housing                         %
%   m2baths    - Combined mass of battery #1 and #2 support housings        %
%   mg         - Mass of gyro                                               %
%   mgh        - Mass of gyro support housing                               %
%   mw(1-3)    - Mass of momentum wheel 1 through 3 (measured)              %
%   mmot(1-3)  - Mass of motor 1 through 3                                  %
%   mamp(1-3)  - Mass of amplifiers                                         %
%   mlx        - Mass of lexan box                                          %
%   msh        - Mass of motor shelves within Lexan Box                     %
%   mcwmech    - Mass of the counterweight mechanism (includes aluminum     %
%                plate structure, three aluminum handknobs, and three       %
%                0.25" threaded steels rods with lengths 17", 12" and 5")   %
%                Note:  The counterweight mechanism plate may not always    %
%                be used.                                                   %
%                                                                          %
%   posb1side  - This is the mass of the positive b1 side.  It will be      %
%                     incremented throughout the plate placement section.  %
%  negb1side   - This is the mass of the negative b1 side.  It will be      %
%                incremented throughout the plate placement section.        %
%                                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear; home;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Structural Component Masses in kg %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

mcsca = 19.3182; mmc1 = 1.603; mmc2 = 1.603; msr1 = 0.579; msr2 =
0.579; msr3 = 0.579; msr4 = 0.579; msr5 = 0.579; msr6 = 0.579;
msr7 = 0.579; msr8 = 0.579; mbp1 = 6.596; mbp2 = 6.596; mmp1 =
3.298; mmp2 = 3.298; mmp3 = 3.298; mmp4 = 3.298; mmp5 = 3.298; mpp
= 3.298;
```

```
posb1side = mmc1+mbp1+msr1+msr2+msr3+msr4; negb1side =
mmc2+mbp2+msr5+msr6+msr7+msr8;

%%%%%%%%%%%%%%%%%%%%%%%%%
% Functional Components %
%%%%%%%%%%%%%%%%%%%%%%%%%

mA = 8.6; mub=0.164; mlb=0.032; mtr = 0.538; mant = 0.3; mbt1 =
5.975; mbt2 = 5.975; m2baths=2.714; mbt3 = 5.975; mbt3hs=1.357;
mbt4 = 0; mbt4hs=0; mg = 1.05; mgh=0.178; mw1 = 2.06; mw2 = 2.06;
mw3 = 2.06; mmot1 = 3.27; mmot2 = 3.27; mmot3 = 3.27; mamp1 =
0.665; mamp2 = 0.665; mamp3 = 0.665; mlx = 5.741; msh = 4.127;
mcwmech= 2.049;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initialize variables--these variables are assigned nonzero values later depending on        %
% whether the user requires a payload plate, a payload, counterweight, or the fine-tuning     %
% mechanism.                                                                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

mcwmechflag1=0; mcwmechflag2=0;

rpp1v=[0;0;0]; payloadv=[0;0;0]; paymass=0; rcwmechv=[0;0;0];
extrav=[0;0;0]; extramass=0;

IpayloadwtC=[0 0 0;0 0 0;0 0 0]; IextrawtC=[0 0 0;0 0 0;0 0 0];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Defining radius vectors from SIMSAT origin (center of central sphere)          %
% to center of mass of each component.                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Structural Components--position vectors (initially given in cm) from      %
%                        the SIMSAT origin                                  %
%                                                                           %
% rcscav - Radius to the centroid of the central sphere/central arm         %
% rmmc1v - Radius to the centroid of the mounting collar +b1 side           %
% rmmc2v - Radius to the centroid of the mounting collar -b1 side           %
% rb1srv - Radius to the centroid of the support rods +b1 side              %
% rb2srv - Radius to the centroid of the support rods -b1 side              %
% rbp1v  - Radius to the centroid of the base plate +b1 side                %
% rbp2v  - Radius to the centroid of the base plate -b1 side                  %
% rmp0v  - Radius to the centroid of the counterweight mechanism plate        %
% rmp1v  - Radius to the centroid of the momentum wheel plate                 %
% rmp2v  - Radius to the centroid of the lexan box plate                      %
% rmp3v  - Radius to the centroid of the two battery plate                    %
% rmp4v  - Radius to the centroid of the battery, gyro, rec/trans plate       %
% rmp5v  - Radius to the centroid of the Autobox plate                        %
% rpp1v   - Radius to the centroid of the payload plate                       %
%                                                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Developing a Method of Adjusting a Component's Position Vector            %
%                                                                           %
% In order to make the code user friendly, we have chosen to minimize the  %
% amount of data entry for the position of every component.  The code is   %
% designed to use the change in position of the MOUNTING PLATES            %
% to calculate the new position vectors of each part (gyro, gyro housing,  %
% battery, battery housing, AutoBox, etc., etc.).                          %
%                                                                          %
% At the time of this coding (March 1999), it is assumed that the relative %
% positions of the components with respect to their mounting plates have   %
% not changed from the baseline design in the SIMSAT thesis.               %
%                                                                          %
% We defined a baseline position for each mounting plate from measurements %
% taken from AutoCAD Drawing "SIMSATFINAL1withobjectgrouping&8RODS.dwg."   %
% A baseline position for the fine-tuning counterweight mechanism plate    %
% was taken from AutoCAD drawing "SIMSATFINAL1nopayldpltcwmechBALANCE.dwg."%
%                                                                          %
% It is assumed the distance from the "b" basis origin (center of the      %
% central sphere) to the center of gravity (cg) of each BASE plate remains %
% fixed.  This base plate distance is defined in rbp1v and rbp2v.          %
% The user only needs to input the measured horizontal distance (in cm)    %
% from the cg of the BASE plate to the cg of a particular mounting plate   %
% (on the same side of SIMSAT as the respective base plate).               %
%                                                                          %
% The values for rmp(0-5)v and rpp1v are determined by adding the user's   %
% measured horizontal distances to rbp1v and rbp2v.  After rmp(0-5)v and   %
% rpp1v are calculated, the change (delta) in the position of the mounting %
% plates from the baseline design is determined.  This position delta      %
% equals the position delta of the individual components on a given        %
% mounting plate.                                                          %
%                                                                          %
% Once the position deltas are calculated, they are added to the baseline  %
% position of the components to give the true component positions with     %
% respect to the "b" basis origin.                                          %
%                                                                          %
%                                                                          %
% comp0delt - This is the position change of the counterweight mechanism   %
%             plate (from the baseline design)                             %
% comp1delt - This is the position change of the momentum wheel plate      %
% comp2delt - This is the position change of the Lexan box plate           %
% comp3delt - This is the position change of the two battery plate         %
% comp4delt - This is the position change of the battery, gyro, t/r plate  %
% comp5delt - This is the position change of the Autobox plate             %
% comp6delt - This is the position change of the payload pegboard plate    %
%                                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


rcscav = [0;0;0]; rmmc1v = [26.93;0;0]; rmmc2v = [-26.93;0;0];
rb1srv = [59.4141;0;0]; rb2srv = [-59.4141;0;0]; rbp1v =
[30.0491;0;0]; rbp2v = [-30.0491;0;0];

u = 'The center of gravity (cg) of each SIMSAT base plate and
mounting plate lies at the middle of its thickness'; disp(u); u =
'dimension. For the following questions, distances are measured
(in cm) from the cg of the base plate to the cg'; disp(u); u = 'of
a particular mounting plate (on the same side of SIMSAT as the
baseplate). When using a measuring tape in the'; disp(u); u =
'SIMSAT laboratory, the user may find it easier to measure the
horizontal "airspace" between the plates and add'; disp(u); u =
'half the thickness of each plate to the "airspace" measurement.
'; disp(u); u = 'Note:  all distances should be entered as
POSITIVE numbers '; disp(u);

u=' '; disp(u);

battplate = input('What is the distance (in cm) from the base
plate to the two battery plate? '); u= ' '; disp(u);

u = 'Press "1" if this plate is on the +b1 side (positive "x"
direction in AutoCAD drawing)'; disp(u); side = input('Otherwise,
press "2". ');

while side ~= 1 & side ~= 2
```

```
    disp('Your input was outside specified range');
    u = 'Enter "1" if this plate is on the +b1 side (positive "x" direction in AutoCAD drawing). ';
    disp(u);
   side = input('Otherwise, enter "2". ');
end

if side == 1
   rmp3v = [rbp1v(1,1)+battplate;0;0];
   posb1side = posb1side+mmp3+mbt1+mbt2+m2baths;
else
   rmp3v = [rbp2v(1,1)-battplate;0;0];
   negb1side = negb1side+mmp3+mbt1+mbt2+m2baths;
end comp3delt = rmp3v(1,1)-42.1175;

u= ' '; disp(u); u = 'The position vector is'; disp(u);
disp(rmp3v); u=input('Press <Enter> to continue:');

home; mwplate = input('What is the distance (in cm) from the base
plate to the momentum wheel plate?'); u = ' '; disp(u); u = 'Press
"1" if this plate is on the +b1 side (positive "x" direction in
AutoCAD drawing)'; disp(u); side = input('Otherwise, press "2".
');

while side ~= 1 & side ~= 2
    disp('Your input was outside specified range');
    u = 'Enter "1" if this plate is on the +b1 side (positive "x" direction in AutoCAD drawing). ';
    disp(u);
   side = input('Otherwise, enter "2". ');
end

if side == 1
   rmp1v = [rbp1v(1,1)+mwplate;0;0];
   posb1side = posb1side+mmp1+mamp1+mamp2+mamp3+mmot1+mmot2+mmot3+mw1+mw2+mw3+msh;
else
   rmp1v = [rbp2v(1,1)-mwplate;0;0];
   negb1side = negb1side+mmp1+mamp1+mamp2+mamp3+mmot1+mmot2+mmot3+mw1+mw2+mw3+msh;
end

comp1delt =rmp1v(1,1)-49.6825;

u= ' '; disp(u); u = 'The position vector is'; disp(u);
disp(rmp1v); u=input('Press <Enter> to continue:');
```

```
home; lexplate = input('What is the distance (in cm) from the base
plate to the Lexan box plate? '); u= ' '; disp(u); if side == 1
   rmp2v = [rbp1v(1,1)+lexplate;0;0];
   posb1side = posb1side+mmp2+mlx;
else
   rmp2v = [rbp2v(1,1)-lexplate;0;0];
   negb1side=negb1side+mmp2+mlx;
end comp2delt = rmp2v(1,1)-77.4351;

u= ' '; disp(u); u = 'The position vector is'; disp(u);
disp(rmp2v); u=input('Press <Enter> to continue:');

home; gyroplate = input('What is the distance (in cm) from the
base plate to the gyro, battery, and transmitter plate? '); u= '
'; disp(u);

u = 'Press "1" if this plate is on the +b1 side (positive "x"
direction in AutoCAD drawing)'; disp(u); side = input('Otherwise,
press "2". ');

while side ~= 1 & side ~= 2
    disp('Your input was outside specified range');
    u = 'Enter "1" if this plate is on the +b1 side (positive "x" direction in AutoCAD drawing). ';
    disp(u);
   side = input('Otherwise, enter "2". ');
end

if side == 1
   rmp4v = [rbp1v(1,1)+gyroplate;0;0];
   posb1side = posb1side+mmp4+mtr+mant+mbt3+mbt3hs+mg+mgh;
else
   rmp4v = [rbp2v(1,1)-gyroplate;0;0];
   negb1side = negb1side+mmp4+mtr+mant+mbt3+mbt3hs+mg+mgh;
end comp4delt = rmp4v(1,1)+43.035;

u= ' '; disp(u); u = 'The position vector is'; disp(u);
disp(rmp4v); u=input('Press <Enter> to continue:');

home; Autoplate = input('What is the distance (in cm) from the
base plate to the AutoBox plate? '); u= ' '; disp(u);

u = 'Press "1" if this plate is on the +b1 side (positive "x"
direction in AutoCAD drawing)'; disp(u); side = input('Otherwise,
press "2". ');

while side ~= 1 & side ~= 2
    disp('Your input was outside specified range');
    u = 'Enter "1" if this plate is on the +b1 side (positive "x" direction in AutoCAD drawing). ';
    disp(u);
   side = input('Otherwise, enter "2". ');
end

if side == 1
   rmp5v = [rbp1v(1,1)+Autoplate;0;0];
   posb1side = posb1side+mmp5+mA+mub+mlb;
else
   rmp5v = [rbp2v(1,1)-Autoplate;0;0];
   negb1side = negb1side+mmp5+mA+mub+mlb;
end

comp5delt = rmp5v(1,1)+48.9325; u= ' '; disp(u); u = 'The position
vector is'; disp(u); disp(rmp5v); u=input('Press <Enter> to
continue:');

home; u = 'Are you using the payload pegboard plate? '; disp(u);
payplateflag =input('Enter 1 for Yes.  Enter 2 for No ');

while payplateflag ~= 1 & payplateflag ~= 2
    disp('Your input was outside specified range');
    payplateflag =input('Enter 1 for Yes.  Enter 2 for No ');
end


if payplateflag == 1

    payplate = input('What is the distance (in cm) from the base plate to the payload plate? ');
    u= ' ';
    disp(u);
    u = 'Press "1" if this plate is on the +b1 side (positive "x" direction in AutoCAD drawing)';
    disp(u);
    side = input('Otherwise, press "2". ');

    while side ~= 1 & side ~= 2
        disp('Your input was outside specified range');
        u = 'Enter "1" if this plate is on the +b1 side (positive "x" direction in AutoCAD
             drawing)';
        disp(u);
     side = input('Otherwise, enter "2". ');
     end

     if side == 1
       rpp1v = [rbp1v(1,1)+payplate;0;0];
       posb1side = posb1side+mpp;
     else
       rpp1v = [rbp2v(1,1)-payplate;0;0];
       negb1side = negb1side+mpp;
   end

   comppdelt = rpp1v(1,1)+72.0675;
    u= ' ';
    disp(u);
    u = 'The position vector is';
    disp(u);
    disp(rpp1v);
   u=input('Press <Enter> to continue:');

else
   mpp=0;
end



home;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Functional Components-position vectors, initially given in cm, FROM the "b" basis %
%                       origin TO the cg of the component                            %
%                                                                                    %
% rAv        - Radius to the centroid of the Autobox                                 %
% rAuv       - Radius to the centroid of the Autobox U-bracket                       %
% rAlv       - Radius to the centroid of the Autobox L-brackets                      %
% rantrv         - Radius to the centroid of the antenna and xmtr/rcvr               %
% rbt3v      - Radius to the centroid of battery 3                                   %
% rbt3hsv    - Radius to the centroid of battery 3 housing                           %
% rgv            - Radius to the centroid of the gyro                                %
% rghv       - Radius to the centroid of gyro housing                                %
% r2batv     - Radius to the centroid of the two batteries                           %
% r2bathsv   - Radius to the centroid of the battery housings                        %
% ramps          - Radius to the centroid of the three amps                          %
% rlxv       - Radius to the centroid of the lexan box                               %
% rshv       - Radius to the centroid of the momentum wheel shelves                 %
% rmotv      - Radius to the centroid of the motors                                 %
% rmw1v      - Radius to the centroid of momentum wheel 1                           %
% rmw2v      - Radius to the centroid of momentum wheel 2                           %
% rmw3v      - Radius to the centroid of momentum wheel 3                           %
%                                                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

rAv = [-59+comp5delt;0;0]; rAuv = [-61.969+comp5delt;0;0]; rAlv =
[-50.5731+comp5delt;0;0]; rantrv = [-39.611+comp4delt;-18.7611;0];
rbt3v = [-38.1625+comp4delt;0;0]; rbt3hsv =
[-37.0623+comp4delt;0.3319;-1.8028]; rgv =
[-38.3+comp4delt;18.6;0]; rghv = [-41.4615+comp4delt;18.1620;0];
r2batv = [37.465+comp3delt;0;0]; r2bathsv =
[38.5651+comp3delt;0;0]; rampsv = [48.0608+comp1delt;0;0]; rlxv  =
[63.3721+comp1delt;0;-1.7922]; rshv  =
[59.1033+comp1delt;-1.3287;-2.2075]; rmotv =
[62.9051+comp1delt;1.4268;0.1284]; rmw1v =
[71.35+comp1delt;-5.735;2.97]; rmw2v =
[62.9+comp1delt;12.715;-11.3]; rmw3v =
[62.8999+comp1delt;5.735;17.1499];

%Converting to meters%
rcscav=rcscav/100; rmmc1v=rmmc1v/100; rmmc2v=rmmc2v/100;
rb1srv=rb1srv/100; rb2srv=rb2srv/100; rbp1v=rbp1v/100;
rbp2v=rbp2v/100; rmp1v=rmp1v/100; rmp2v=rmp2v/100;
rmp3v=rmp3v/100; rmp4v=rmp4v/100; rmp5v=rmp5v/100;
rpp1v=rpp1v/100; rAv=rAv/100; rAuv=rAuv/100; rAlv=rAlv/100;
rantrv=rantrv/100; rbt3v=rbt3v/100; rbt3hsv=rbt3hsv/100;
rgv=rgv/100; rghv=rghv/100; r2batv=r2batv/100;
r2bathsv=r2bathsv/100; rampsv=rampsv/100; rlxv=rlxv/100;
rshv=rshv/100; rmotv=rmotv/100; rmw1v=rmw1v/100; rmw2v=rmw2v/100;
rmw3v=rmw3v/100;



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Section 2 -- Counterweight is positioned using center of mass calculations     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

massdifference = posb1side - negb1side; if massdifference < 0
   disp('The -b1 side has more mass than the +b1 side.  The difference, in kg, is:');
   disp(massdifference);
   diffflag=1;
else
   disp('The +b1 side has more mass than the -b1 side.  The difference, in kg, is:');
   disp(massdifference);
   diffflag=2;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Here we calculate the center of SIMSAT WITHOUT the payload, counterweight,     %
% or fine-tuning counterweight mechanism                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

num =
mcsca*rcscav+mmc1*rmmc1v+mmc2*rmmc2v+(msr1+msr2+msr3+msr4)*rb1srv+(msr5+msr6+msr7+msr8)*rb2srv...
+mbp1*rbp1v+mbp2*rbp2v+mmp1*rmp1v+mmp2*rmp2v+mmp3*rmp3v+mmp4*rmp4v+mmp5*rmp5v+mpp*rpp1v...
+mA*rAv+mub*rAuv+mlb*rAlv+(mant+mtr)*rantrv+mbt3*rbt3v+mbt3hs*rbt3hsv+mg*rgv+mgh*rghv...
+(mbt1+mbt2)*r2batv+m2baths*r2bathsv+(mamp1+mamp2+mamp3)*rampsv+(mmot1+mmot2+mmot3)*rmotv...
+mw1*rmw1v+mw2*rmw2v+mw3*rmw3v+mlx*rlxv+msh*rshv; den =
mcsca+mmc1+mmc2+msr1+msr2+msr3+msr4+msr5+msr6+msr7+msr8+mbp1+mbp2+mmp1+mmp2+mmp3+mmp4+mmp5+mpp+mA...
+mub+mlb+mant+mtr+mbt3+mbt3hs+mg+mgh+mbt1+mbt2+m2baths+mamp1+mamp2+mamp3+mmot1+mmot2+mmot3+mw1...
+mw2+mw3+mlx+msh;

u=' '; disp(u); u='The composite SIMSAT center of mass (in cm),
before adding the payload or counterweight mechanism, is located
at: '; disp(u); rcomp = (num*(1/den));
rcompcm=rcomp*100; %rcomp in cm
disp(rcompcm);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Here we ask the user if he wants to GROSS balance SIMSAT by attaching a payload or %
% counterweight blocks to the payload pegboard plate                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if payplateflag==1
    u='As a theoretical reference, the following mass (in kg), which has its cg colocated with
       the cg of the ';
    disp(u);
    u='payload mounting plate, will balance SIMSAT about the b1 axis: ';
    disp(u);

    suggestedcw= -(mcsca*rcscav(1,1)+mmc1*rmmc1v(1,1)+mmc2*rmmc2v(1,1)...
   +(msr1+msr2+msr3+msr4)*rb1srv(1,1)+(msr5+msr6+msr7+msr8)*rb2srv(1,1)+mbp1*rbp1v(1,1)...
   +mbp2*rbp2v(1,1)+mmp1*rmp1v(1,1)+mmp2*rmp2v(1,1)+mmp3*rmp3v(1,1)+mmp4*rmp4v(1,1)...
+mmp5*rmp5v(1,1)+mpp*rpp1v(1,1)+mA*rAv(1,1)+mub*rAuv(1,1)+mlb*rAlv(1,1)+(mant+mtr)*rantrv(1,1)...
+mbt3*rbt3v(1,1)+mbt3hs*rbt3hsv(1,1)+mg*rgv(1,1)+mgh*rghv(1,1)+(mbt1+mbt2)*r2batv(1,1)...
+m2baths*r2bathsv(1,1)+(mamp1+mamp2+mamp3)*rampsv(1,1)+(mmot1+mmot2+mmot3)*rmotv(1,1)...
+mw1*rmw1v(1,1)+mw2*rmw2v(1,1)+mw3*rmw3v(1,1)+mlx*rlxv(1,1)+msh*rshv(1,1))/rpp1v(1,1);

 disp(suggestedcw);
 u=' ';
 disp(u);
 u = 'With the theoretical reference mass in mind and if you are using the payload
      pegboard plate, ';
 disp(u);
 u= 'do you want to attach a payload or counterweight blocks to this plate? ';
 disp(u);
 payloadflag =input('Enter 1 for Yes.  Enter 2 for No ');

 while payloadflag ~= 1 & payloadflag ~= 2
     disp('Your input was outside specified range');
     payloadflag =input('Enter 1 for Yes.  Enter 2 for No ');
 end

 while payloadflag==1
     paymass=input('What is the total mass (in kg) of the payload or counterweight blocks
                    you wish to add? ');
     disp(' ');
     disp('The cg (in cm) of the payload mounting plate (with respect to the "b" basis ORIGIN)
          is currently at');
     disp(rpp1v*100);

     disp(' ')
     u='With respect to the "b" basis ORIGIN (located at the center of the central sphere), where
        is the cg (in cm) of';
     disp(u);
     u='your payload and/or counterweight blocks? ';
     disp(u);
     payloadv(1,1)=input('Enter b1 vector component in cm ');
     payloadv(2,1)=input('Enter b2 vector component in cm ');
     payloadv(3,1)=input('Enter b3 vector component in cm ');
     payloadv=payloadv/100; %convert to meters

     u=' ';
     disp(u);
     u='With your payload and/or counterweight blocks INCLUDED, the composite SIMSAT
        center of mass';
```

```
    disp(u);
    u='(in cm) is now located at:';
  disp(u);
  disp(' ');

    num = mcsca*rcscav+mmc1*rmmc1v+mmc2*rmmc2v+(msr1+msr2+msr3+msr4)*rb1srv...
    +(msr5+msr6+msr7+msr8)*rb2srv+mbp1*rbp1v+mbp2*rbp2v+mmp1*rmp1v+mmp2*rmp2v+mmp3*rmp3v...
    +mmp4*rmp4v+mmp5*rmp5v+mpp*rpp1v+mA*rAv+mub*rAuv+mlb*rAlv+(mant+mtr)*rantrv+mbt3*rbt3v...
    +mbt3hs*rbt3hsv+mg*rgv+mgh*rghv+(mbt1+mbt2)*r2batv+m2baths*r2bathsv...
    +(mamp1+mamp2+mamp3)*rampsv+(mmot1+mmot2+mmot3)*rmotv+mw1*rmw1v+mw2*rmw2v+mw3*rmw3v...
    +mlx*rlxv+msh*rshv+paymass*payloadv;

    den = mcsca+mmc1+mmc2+msr1+msr2+msr3+msr4+msr5+msr6+msr7+msr8+mbp1+mbp2+mmp1+mmp2+mmp3...
    +mmp4+mmp5+mpp+mA+mub+mlb+mant+mtr+mbt3+mbt3hs+mg+mgh+mbt1+mbt2+m2baths+mamp1+mamp2...
    +mamp3+mmot1+mmot2+mmot3+mw1+mw2+mw3+mlx+msh+paymass;

    rcomp = (num*(1/den));
  rcompcm = rcomp*100; %rcomp in cm
  disp(rcompcm);

    u='Would you like to change your payload mass or location of the payload cg to change
       "rcomp" above? ';
    disp(u);
    payloadflag=input('Enter 1 for Yes.  Enter 2 for No.');
    while payloadflag ~= 1 & payloadflag ~= 2
        disp('Your input was outside specified range');
        payloadflag =input('Enter 1 for Yes.  Enter 2 for No ');
    end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Determine the inertia (with respect to the composite center of mass)of the        %
% payload/counterweight block here--this will only be used if the user chooses      %
% this option later in the program.  Payload/counterweight block is assumed to be a %
% 2" by variable" by 4" rectangular carbon steel block                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

rpayloadwtC=payloadv-rcomp;

u1cw=.0254*2; %b1 dimension
u3cw=.0254*4; %b3 dimension
u2cw=paymass/(u1cw*u3cw*7870); %density of carbon steel = 7870 kg/m^3
u2cwcm=u2cw*100; %b2 dimension in cm
```

```
    disp(u2cwcm);

    Icwcw1(1,1) = paymass*(u2cw^2+u3cw^2)/12;
    Icwcw1(1,2) = 0;
    Icwcw1(1,3) = 0;
    Icwcw1(2,1) = 0;
    Icwcw1(2,2) = paymass*(u1cw^2+u3cw^2)/12;
    Icwcw1(2,3) = 0;
    Icwcw1(3,1) = 0;
    Icwcw1(3,2) = 0;
    Icwcw1(3,3) = paymass*(u1cw^2+u2cw^2)/12;

    %u = 'Payload/Counterweight Block Inertia Matrix (about payload/counterweight block mass
    %     center)';
    %disp(u);
    Icwcw1;
    paraxispayloadwt=[rpayloadwtC(2)^2+rpayloadwtC(3)^2 -rpayloadwtC(1)*rpayloadwtC(2)...
     -rpayloadwtC(1)*rpayloadwtC(3);-rpayloadwtC(1)*rpayloadwtC(2)...
     rpayloadwtC(1)^2+rpayloadwtC(3)^2 -rpayloadwtC(2)*rpayloadwtC(3);...
     -rpayloadwtC(1)*rpayloadwtC(3) -rpayloadwtC(2)*rpayloadwtC(3)...
     rpayloadwtC(1)^2+rpayloadwtC(2)^2];
    IpayloadwtC=Icwcw1+paymass*paraxispayloadwt;

    end

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Here we ask the user if he wants to account for the mass balancing effects of    %
% the fine-tuning counterweight mechanism                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

home; u=' '; u='The current SIMSAT center of mass, in cm, is
located at: '; disp(u); disp(' ');

num =
mcsca*rcscav+mmc1*rmmc1v+mmc2*rmmc2v+(msr1+msr2+msr3+msr4)*rb1srv+(msr5+msr6+msr7+msr8)*rb2srv...
        +mbp1*rbp1v+mbp2*rbp2v+mmp1*rmp1v+mmp2*rmp2v+mmp3*rmp3v+mmp4*rmp4v+mmp5*rmp5v+mpp*rpp1v...
        +mA*rAv+mub*rAuv+mlb*rAlv+(mant+mtr)*rantrv+mbt3*rbt3v+mbt3hs*rbt3hsv+mg*rgv+mgh*rghv...
        +(mbt1+mbt2)*r2batv+m2baths*r2bathsv+(mamp1+mamp2+mamp3)*rampsv...
        +(mmot1+mmot2+mmot3)*rmotv+mw1*rmw1v+mw2*rmw2v+mw3*rmw3v+mlx*rlxv+msh*rshv+paymass*payloadv;

den =
mcsca+mmc1+mmc2+msr1+msr2+msr3+msr4+msr5+msr6+msr7+msr8+mbp1+mbp2+mmp1+mmp2+mmp3+mmp4+mmp5+mpp+mA...
+mub+mlb+mant+mtr+mbt3+mbt3hs+mg+mgh+mbt1+mbt2+m2baths+mamp1+mamp2+mamp3+mmot1+mmot2+mmot3+mw1...
+mw2+mw3+mlx+msh+paymass;

rcomp = (num*(1/den));
rcompcm = rcomp*100; %rcomp in cm

disp(rcompcm);


u=' '; disp(u); u = 'Would you like to use the fine-tuning
counterweight mechanism plate? '; disp(u); cwmechflag
=input('Enter 1 for Yes.  Enter 2 for No ');

while cwmechflag ~= 1 & cwmechflag ~= 2
    disp('Your input was outside specified range');
    cwmechflag =input('Enter 1 for Yes.  Enter 2 for No ');
end

if cwmechflag == 1
    cwmechplate = input('What is the distance (in cm) from the base plate to the counterweight
                        mechanism plate? ');
    u= ' ';
    disp(u);
    u = 'Press "1" if this plate is on the +b1 side (positive "x" direction in AutoCAD drawing)';
    disp(u);
    side = input('Otherwise, press "2". ');

    while side ~= 1 & side ~= 2
        disp('Your input was outside specified range');
        u = 'Enter "1" if this plate is on the +b1 side (positive "x" direction in AutoCAD
             drawing). ';
        disp(u);
    side = input('Otherwise, enter "2". ');
    end

    if side == 1
      rmp0v = [rbp1v(1,1)*100+cwmechplate;0;0];
      mcwmechflag1=mcwmech;
    else
      rmp0v = [rbp2v(1,1)*100-cwmechplate;0;0];
      mcwmechflag2=mcwmech;
   end
```

```
compOdelt = rmpOv(1,1)-80.8143;
 u= ' ';
 disp(u);
 u = 'The position vector (in cm) of the fine-tuning PLATE (does NOT include rods, knobs,
      etc.) is';
 disp(u);
 disp(rmpOv);
u=input('Press <Enter> to continue:');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Account for actual center of mass of the BARE cw fine-tuning mechanism.  The actual %
% center of mass was calculated in MathCad (in "centerofmassforcwmechnism.mcd") by    %
% assuming the 17" and 12" threaded steels rods were centered on the plate. The 5"    %
% threaded rod (b1 axis rod) was assumed to be tightened until it was flush with the  %
% inside surface of the plate.  The masses of the small aluminum handknobs were also  %
% accounted for.                                                                      %
%                                                                                     %
% rcwmechvcm - position vector FROM the center of the sphere TO the centroid of       %
%              the cw mechanism in cm                                                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 rcwmechvcm = [78.39879+compOdelt;0.13583;0.09588]; %this is in cm
 rcwmechv=rcwmechvcm/100; %convert to meters

 u=' ';
 disp(u);
u='The composite SIMSAT center of mass (in cm), with the BARE fine-tuning MECHANISM
  (i.e., plate, ';
disp(u);
u='3 threaded rods, rod holders, and 3 handknobs) included, is now: ';
disp(u);
disp(' ');

 num = mcsca*rcscav+mmc1*rmmc1v+mmc2*rmmc2v+(msr1+msr2+msr3+msr4)*rb1srv...
 +(msr5+msr6+msr7+msr8)*rb2srv+mbp1*rbp1v+mbp2*rbp2v+mmp1*rmp1v+mmp2*rmp2v+mmp3*rmp3v...
 +mmp4*rmp4v+mmp5*rmp5v+mpp*rpp1v+mA*rAv+mub*rAuv+mlb*rAlv+(mant+mtr)*rantrv+mbt3*rbt3v...
 +mbt3hs*rbt3hsv+mg*rgv+mgh*rghv+(mbt1+mbt2)*r2batv+m2baths*r2bathsv...
 +(mamp1+mamp2+mamp3)*rampsv+(mmot1+mmot2+mmot3)*rmotv+mw1*rmw1v+mw2*rmw2v+mw3*rmw3v...
 +mlx*rlxv+msh*rshv+paymass*payloadv+mcwmech*rcwmechv;

 den = mcsca+mmc1+mmc2+msr1+msr2+msr3+msr4+msr5+msr6+msr7+msr8+mbp1+mbp2+mmp1+mmp2+mmp3+mmp4...
 +mmp5+mpp+mA+mub+mlb+mant+mtr+mbt3+mbt3hs+mg+mgh+mbt1+mbt2+m2baths+mamp1+mamp2+mamp3+mmot1...
 +mmot2+mmot3+mw1+mw2+mw3+mlx+msh+paymass+mcwmech;


 rcomp = (num*(1/den));
rcompcm = rcomp*100; %rcomp in cm
disp(rcompcm);
disp(' ');


 u='Would you like to add mass to the fine-tuning mechanism to change "rcomp" above? ';
 disp(u);
 continueflag=input('Enter 1 for Yes.  Enter 2 for No.');
 while continueflag ~= 1 & continueflag ~= 2
     disp('Your input was outside specified range');
     continueflag =input('Enter 1 for Yes.  Enter 2 for No ');
 end


 while continueflag==1
 disp(' ');
 extramass=input('What is the total mass (in kg) of the weights to be added to the fine-tuning
                 mechanism ');
 disp(' ');


 disp('The cg (in cm) of the BARE fine-tuning MECHANISM (with respect to the "b" basis ORIGIN)
      is currently at:');
   disp(' ');
   disp(rcwmechvcm);


 disp(' ');
     u='With respect to the "b" basis ORIGIN (located at the center of the central sphere),
        where is the cg (in cm) of';
     disp(u);
     u='your additional weights? ';
   disp(u);
   disp(' ');
     extrav(1,1)=input('Enter b1 vector component in cm ');
     extrav(2,1)=input('Enter b2 vector component in cm ');
     extrav(3,1)=input('Enter b3 vector component in cm ');
     extrav=extrav/100; %convert to meters


     disp(' ');
     u='With your additional fine-tuning weights INCLUDED, the composite SIMSAT center of mass';
     disp(u);
     u='(in cm) is now located at:';
   disp(u);
   disp(' ');

    num = mcsca*rcscav+mmc1*rmmc1v+mmc2*rmmc2v+(msr1+msr2+msr3+msr4)*rb1srv...
    +(msr5+msr6+msr7+msr8)*rb2srv+mbp1*rbp1v+mbp2*rbp2v+mmp1*rmp1v+mmp2*rmp2v+mmp3*rmp3v...
    +mmp4*rmp4v+mmp5*rmp5v+mpp*rpp1v+mA*rAv+mub*rAuv+mlb*rAlv+(mant+mtr)*rantrv+mbt3*rbt3v...
    +mbt3hs*rbt3hsv+mg*rgv+mgh*rghv+(mbt1+mbt2)*r2batv+m2baths*r2bathsv...
    +(mamp1+mamp2+mamp3)*rampsv+(mmot1+mmot2+mmot3)*rmotv+mw1*rmw1v+mw2*rmw2v+mw3*rmw3v...
    +mlx*rlxv+msh*rshv+paymass*payloadv+mcwmech*rcwmechv+extramass*extrav;

    den = mcsca+mmc1+mmc2+msr1+msr2+msr3+msr4+msr5+msr6+msr7+msr8+mbp1+mbp2+mmp1+mmp2+mmp3...
    +mmp4+mmp5+mpp+mA+mub+mlb+mant+mtr+mbt3+mbt3hs+mg+mgh+mbt1+mbt2+m2baths+mamp1+mamp2...
    +mamp3+mmot1+mmot2+mmot3+mw1+mw2+mw3+mlx+msh+paymass+mcwmech+extramass;

    rcomp = (num*(1/den));
  rcompcm = rcomp*100; %rcomp in cm
  disp(rcompcm);

    disp(' ');
    u ='Would you like to change your fine-tuning mass or location of the fine-tuning cg to
        change "rcomp" above? ';
    disp(u);
  continueflag=input('Enter 1 for Yes.  Enter 2 for No.');
  disp(' ');
    while continueflag ~= 1 & continueflag ~= 2
        disp('Your input was outside specified range');
        continueflag =input('Enter 1 for Yes.  Enter 2 for No ');
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Determine the inertia (with respect to the composite center of mass) of the  %
% user-added fine-tuning mass here--this will only be used if the user chooses %
% this option later in the program.  Fine-tuning mass is assumed to be a SINGLE %
% 2" by variable" by 4" rectangular carbon steel block.                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

rextrawtC=extrav-rcomp;

u1extra=.0254*2; %b1 dimension
u3extra=.0254*4; %b3 dimension
u2extra=extramass/(u1extra*u3extra*7870); %density of carbon steel = 7870 kg/m^3
u2extracm=u2extra*100; %b2 dimension in cm
```

```
     Iextra1(1,1) = extramass*(u2extra^2+u3extra^2)/12;
     Iextra1(1,2) = 0;
     Iextra1(1,3) = 0;
     Iextra1(2,1) = 0;
     Iextra1(2,2) = extramass*(u1extra^2+u3extra^2)/12;
     Iextra1(2,3) = 0;
     Iextra1(3,1) = 0;
     Iextra1(3,2) = 0;
     Iextra1(3,3) = extramass*(u1extra^2+u2extra^2)/12;

     %u = 'Fine-tuning Mass Block Inertia Matrix (about fine-tuning block mass center)';
     %disp(u);
     Iextra1;
     paraxisextrawt=[rextrawtC(2)^2+rextrawtC(3)^2 -rextrawtC(1)*rextrawtC(2)...
      -rextrawtC(1)*rextrawtC(3);-rextrawtC(1)*rextrawtC(2) rextrawtC(1)^2+rextrawtC(3)^2...
      -rextrawtC(2)*rextrawtC(3);-rextrawtC(1)*rextrawtC(3) -rextrawtC(2)*rextrawtC(3)...
      rextrawtC(1)^2+rextrawtC(2)^2];
     IextrawtC=Iextra1+extramass*paraxisextrawt;

else
   mcwmech=0;

end %end if statement

u=input('Press <Enter> to continue.');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Now we are computing the radius vector, rcomp, from the Simsat origin  %
% the composite center of mass to doublecheck our calculations.          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

num =
mcsca*rcscav+mmc1*rmmc1v+mmc2*rmmc2v+(msr1+msr2+msr3+msr4)*rb1srv+(msr5+msr6+msr7+msr8)*rb2srv...
+mbp1*rbp1v+mbp2*rbp2v+mmp1*rmp1v+mmp2*rmp2v+mmp3*rmp3v+mmp4*rmp4v+mmp5*rmp5v+mpp*rpp1v+mA*rAv...
+mub*rAuv+mlb*rAlv+(mant+mtr)*rantrv+mbt3*rbt3v+mbt3hs*rbt3hsv+mg*rgv+mgh*rghv...
+(mbt1+mbt2)*r2batv+m2baths*r2bathsv+(mamp1+mamp2+mamp3)*rampsv+(mmot1+mmot2+mmot3)*rmotv...
+mw1*rmw1v+mw2*rmw2v+mw3*rmw3v+mlx*rlxv+msh*rshv+paymass*payloadv+mcwmech*rcwmechv+extramass*extrav;

den =
mcsca+mmc1+mmc2+msr1+msr2+msr3+msr4+msr5+msr6+msr7+msr8+mbp1+mbp2+mmp1+mmp2+mmp3+mmp4+mmp5+mpp+mA...
+mub+mlb+mant+mtr+mbt3+mbt3hs+mg+mgh+mbt1+mbt2+m2baths+mamp1+mamp2+mamp3+mmot1+mmot2+mmot3+mw1...
+mw2+mw3+mlx+msh+paymass+mcwmech+extramass;
```

```
rcomp = (num*(1/den)); %this is in meters


u='The SIMSAT center of mass, rcomp, in cm is: '; disp(u); disp('
'); rcompcm=rcomp*100; disp(rcompcm);


mssat=den; home;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Section 3 -- Calculate inertia matrices of the wheels with respect to their own   %
%              centers of mass and bases, with respect to their own centers of mass %
%              in the "b" basis, and with respect to the SIMSAT center of mass in   %
%              the "b" basis                                                        %
%                                                                                   %
% Calculating the inertia matrices for the momentum wheels is necessary             %
% because the SIMULINK and MATLAB motion simulation programs require the            %
% wheel inertia matrices                                                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


  densityhoop=7.85; %density of steel in g/cc
  densitydisk=2.8;  %density of aluminum in g/cc



  r1o=10.95375;
  wrim1=.9525;
  t1=3.0111;
  tdisk1=.635;


  %% r1o=outer radius of steel hoop  in cm         %%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %% wrim1=width of steel hoop in cm               %%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %% t1=thickness (height) of  steel hoop in cm    %%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %% tdisk1=thickness of thin aluminum disk in cm  %%%%%%%%%%%%%%%%%%%%%%%%%%%%


  r1i=r1o-wrim1;
  m1o=(densityhoop*pi*r1o^2*t1)/1000;
  m1i=(densityhoop*pi*r1i^2*t1)/1000;
  mhoop1=m1o-m1i;
  mdisk1=(densitydisk*pi*r1i^2*tdisk1)/1000;
  m1=mhoop1+mdisk1; %Solid steel hoop case %%%%%%%%%%%%%%%%%%%%%%%%


  m2o=m1o;
  m3o=m1o;
  m2i=m1i;
  m3i=m1i;
  r2o=r1o;
  r3o=r1o;
  r2i=r1i;
  r3i=r1i;
  t2=t1;
  t3=t1;
  tdisk2=tdisk1;
  tdisk3=tdisk1;
  m2=m1;
  m3=m1;
  mhoop2=mhoop1;
  mhoop3=mhoop1;
  mdisk2=mdisk1;
  mdisk3=mdisk1;

%Convert r and t to meters
  r1o=r1o/100;
  r1i=r1i/100;
  t1=t1/100;
  tdisk1=tdisk1/100;
  r2o=r2o/100;
  r2i=r2i/100;
  t2=t2/100;
  tdisk2=tdisk2/100;
  r3o=r3o/100;
  r3i=r3i/100;
  t3=t3/100;
  tdisk3=tdisk3/100;

%Calculate inertia matrix for thin aluminum disk
  J11=(3*r1i^2+tdisk1^2)/12;
  J12=(r1i^2)/2;
  J13=(3*r1i^2+tdisk1^2)/12;
  Jdisk1=mdisk1*[J11 0 0; 0 J12 0; 0 0 J13];

  J21=(3*r2i^2+tdisk2^2)/12;
  J22=(r2i^2)/2;
  J23=(3*r2i^2+tdisk2^2)/12;
  Jdisk2=mdisk2*[J21 0 0; 0 J22 0; 0 0 J23];

  J31=(3*r3i^2+tdisk3^2)/12;
  J32=(r3i^2)/2;
  J33=(3*r3i^2+tdisk3^2)/12;
  Jdisk3=mdisk3*[J31 0 0; 0 J32 0; 0 0 J33];

  %Calculating inertia matrix for outer radius of steel hoop
  Jo11=(3*r1o^2+t1^2)/12;
  Jo12=(r1o^2)/2;
  Jo13=(3*r1o^2+t1^2)/12;
  Johoop1=[Jo11 0 0; 0 Jo12 0; 0 0 Jo13];

  Jo21=(3*r2o^2+t2^2)/12;
  Jo22=(r2o^2)/2;
  Jo23=(3*r2o^2+t2^2)/12;
  Johoop2=[Jo21 0 0; 0 Jo22 0; 0 0 Jo23];

  Jo31=(3*r3o^2+t3^2)/12;
  Jo32=(r3o^2)/2;
  Jo33=(3*r3o^2+t3^2)/12;
  Johoop3=[Jo31 0 0; 0 Jo32 0; 0 0 Jo33];

  %Calculating inertia matrix for inner radius of steel hoop
  Ji11=(3*r1i^2+t1^2)/12;
  Ji12=(r1i^2)/2;
  Ji13=(3*r1i^2+t1^2)/12;
  Jihoop1=[Ji11 0 0; 0 Ji12 0; 0 0 Ji13];

  Ji21=(3*r2i^2+t2^2)/12;
  Ji22=(r2i^2)/2;
  Ji23=(3*r2i^2+t2^2)/12;
  Jihoop2=[Ji21 0 0; 0 Ji22 0; 0 0 Ji23];

  Ji31=(3*r3i^2+t3^2)/12;
  Ji32=(r3i^2)/2;
  Ji33=(3*r3i^2+t3^2)/12;
  Jihoop3=[Ji31 0 0; 0 Ji32 0; 0 0 Ji33];

  %Calculating inertia matrix for the steel hoop
  Jhoop1 = m1o*Johoop1 - m1i*Jihoop1;
  Jhoop2 = m2o*Johoop2 - m2i*Jihoop2;
  Jhoop3 = m3o*Johoop3 - m3i*Jihoop3;

%Calculate inertia matrix for entire wheel
  Jwheel1=Jdisk1+Jhoop1;
  Jwheel2=Jdisk2+Jhoop2;
  Jwheel3=Jdisk3+Jhoop3;
```

```
%Note:  rotaw1 is wheel 1 rotation matrix going one way, 1-2-3
%User will input angles for rotation matrices to orient momentum wheels

u=['Initial orientation of all momentum wheels is with their
second axis (axis of rotation)'];
%disp(u);
u=['pointing along respective Simsat "b" axis.  The following
rotations will align wheel axes with Simsat axes:'];
%disp(u);

% Momentum Wheel 1 %
thetarot1w1=0;%input('How many degrees do you want to rotate wheel 1 about its 1st axis? ');
thetarot2w1=0;%input('How many degrees do you want to rotate wheel 1 about its new 2nd axis? ');
thetarot3w1=90;%input('How many degrees do you want to rotate wheel 1 about its new 3rd axis? ');

thetarot1w1rad=thetarot1w1*pi/180;
thetarot2w1rad=thetarot2w1*pi/180;
thetarot3w1rad=thetarot3w1*pi/180;

rot1w1=[1 0 0;0 cos(thetarot1w1rad) sin(thetarot1w1rad);0
-sin(thetarot1w1rad) cos(thetarot1w1rad)];
rot2w1=[cos(thetarot2w1rad) 0 -sin(thetarot2w1rad);0 1
0;sin(thetarot2w1rad) 0 cos(thetarot2w1rad)];
rot3w1=[cos(thetarot3w1rad) sin(thetarot3w1rad)
0;-sin(thetarot3w1rad) cos(thetarot3w1rad) 0;0 0 1];

rotaw1=rot3w1*(rot2w1*rot1w1); rotbw1=rotaw1'; Jw1w1d=Jwheel1;
Jw1w1b=rotaw1*(Jw1w1d*rotbw1);

% Momentum Wheel 2 %
thetarot1w2=0;%input('Moving on to wheel 2.  How many degrees do you want to rotate wheel 2 about
              %its 1st axis? ');
thetarot2w2=0;%input('How many degrees do you want to rotate wheel 2 about its new 2nd axis?');
thetarot3w2=0;%input('How many degrees do you want to rotate wheel 2 about its new 3rd axis?');

thetarot1w2rad=thetarot1w2*pi/180;
thetarot2w2rad=thetarot2w2*pi/180;
thetarot3w2rad=thetarot3w2*pi/180;

rot1w2=[1 0 0;0 cos(thetarot1w2rad) sin(thetarot1w2rad);0
-sin(thetarot1w2rad) cos(thetarot1w2rad)];
rot2w2=[cos(thetarot2w2rad) 0 -sin(thetarot2w2rad);0 1
```

```
0;sin(thetarot2w2rad) 0 cos(thetarot2w2rad)];
rot3w2=[cos(thetarot3w2rad) sin(thetarot3w2rad)
0;-sin(thetarot3w2rad) cos(thetarot3w2rad) 0;0 0 1];

rotaw2=rot3w2*(rot2w2*rot1w2); rotbw2=rotaw2'; Jw2w2f=Jwheel2;
Jw2w2b=rotaw2*(Jw2w2f*rotbw2);


% Momentum Wheel 3 %
thetarot1w3=-90;%input('Finally, enter data for wheel 3.  How many degrees do you want to rotate
                %wheel 3 about its 1st axis?');
thetarot2w3=0;%input('How many degrees do you want to rotate wheel 3 about its new 2nd axis?');
thetarot3w3=0;%input('How many degrees do you want to rotate wheel 3 about its new 3rd axis?');

thetarot1w3rad=thetarot1w3*pi/180;
thetarot2w3rad=thetarot2w3*pi/180;
thetarot3w3rad=thetarot3w3*pi/180;

rot1w3=[1 0 0;0 cos(thetarot1w3rad) sin(thetarot1w3rad);0
-sin(thetarot1w3rad) cos(thetarot1w3rad)];
rot2w3=[cos(thetarot2w3rad) 0 -sin(thetarot2w3rad);0 1
0;sin(thetarot2w3rad) 0 cos(thetarot2w3rad)];
rot3w3=[cos(thetarot3w3rad) sin(thetarot3w3rad)
0;-sin(thetarot3w3rad) cos(thetarot3w3rad) 0;0 0 1];

rotaw3=rot3w3*(rot2w3*rot1w3); rotbw3=rotaw3'; Jw3w3h=Jwheel3;
Jw3w3b=rotaw3*(Jw3w3h*rotbw3);

u = 'Momentum Wheel Inertia Matrices (Momentum wheel mass
center)';
%disp(u);
Jw1w1b; Jw2w2b; Jw3w3b;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Momentum Wheel Inertia Matrices %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

rvectorwheel1=rmw1v-rcomp; rvectorwheel2=rmw2v-rcomp;
rvectorwheel3=rmw3v-rcomp;

paraxis1=[rvectorwheel1(2)^2+rvectorwheel1(3)^2
-rvectorwheel1(1)*rvectorwheel1(2)
```

```
-rvectorwheel1(1)*rvectorwheel1(3);-rvectorwheel1(1)*rvectorwheel1(2)
rvectorwheel1(1)^2+rvectorwheel1(3)^2
-rvectorwheel1(2)*rvectorwheel1(3);-rvectorwheel1(1)*rvectorwheel1(3)
-rvectorwheel1(2)*rvectorwheel1(3)
rvectorwheel1(1)^2+rvectorwheel1(2)^2]; Jw1cb=Jw1w1b+mw1*paraxis1;

paraxis2=[rvectorwheel2(2)^2+rvectorwheel2(3)^2
-rvectorwheel2(1)*rvectorwheel2(2)
-rvectorwheel2(1)*rvectorwheel2(3);-rvectorwheel2(1)*rvectorwheel2(2)
rvectorwheel2(1)^2+rvectorwheel2(3)^2
-rvectorwheel2(2)*rvectorwheel2(3);-rvectorwheel2(1)*rvectorwheel2(3)
-rvectorwheel2(2)*rvectorwheel2(3)
rvectorwheel2(1)^2+rvectorwheel2(2)^2]; Jw2cb=Jw2w2b+mw2*paraxis2;

paraxis3=[rvectorwheel3(2)^2+rvectorwheel3(3)^2
-rvectorwheel3(1)*rvectorwheel3(2)
-rvectorwheel3(1)*rvectorwheel3(3);-rvectorwheel3(1)*rvectorwheel3(2)
rvectorwheel3(1)^2+rvectorwheel3(3)^2
-rvectorwheel3(2)*rvectorwheel3(3);-rvectorwheel3(1)*rvectorwheel3(3)
-rvectorwheel3(2)*rvectorwheel3(3)
rvectorwheel3(1)^2+rvectorwheel3(2)^2]; Jw3cb=Jw3w3b+mw3*paraxis3;



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Section 4 -- Autocad inertia matrix is input by the user                         %
%                                                                                  %
% The matrix elements will be filled in by the user.  The matrix from AutoCAD      %
% represents the dimensional inertia in cm^2 and AutoCAD assumes a density         %
% of 1g/cm^3.   Therefore, we must convert to kg*m^2 units and account for the     %
% actual average density of SIMSAT.                                                %
%                                                                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

u='If you do NOT have your payload, counterweight and/or
fine-tuning weight (hollow cylindrical '; disp(u); u='weights on
the threaded rods or small blocks) drawn in AutoCAD, this program
will calculate the '; disp(u); u='inertia matrix (about the SIMSAT
center of mass) for these items. '; disp(u); u='This program
assumes the payload, counterweight and/or fine-tuning weight are
rectangular carbon steel blocks. '; disp(u); u='These steel blocks
are assumed to have the following dimensions: '; disp(u); disp('
'); u='b1 dimension = 5.08 cm (2 in) '; disp(u); u='b2 dimension =
a variable (calculated by MATLAB) that gives a steel block a
particular mass '; disp(u); u='b3 dimension = 10.16 cm (4 in) ';
disp(u); disp(' ');

u='Do you already have your "payload" accurately drawn in the
SIMSAT AutoCAD drawing? '; disp(u); u='Enter 1 for YES (since the
"payload" is accounted for by AutoCAD, MATLAB will NOT calculate
the payload inertia). '; disp(u); disp(' ');
paymassflag=input('Enter 2 for NO (MATLAB will calculate the
"payload" inertia from a rectangular carbon steel block).');

while paymassflag ~= 1 & paymassflag ~= 2
    disp('Your input was outside specified range');
   paymassflag =input('Enter 1 for Yes.  Enter 2 for No ');
end if paymassflag==2
   mssat=mssat-paymass;
end

disp(' '); disp(' '); u='Do you already have the fine-tuning
weights (hollow cylindrical weights on the threaded rods or small
blocks)'; disp(u); u='accurately drawn in the SIMSAT AutoCAD
drawing? '; disp(u); finemassflag=input('Enter 1 for YES. Enter 2
for NO. '); while finemassflag ~= 1 & finemassflag ~= 2
    disp('Your input was outside specified range');
    finemassflag =input('Enter 1 for Yes.  Enter 2 for No ');
end if finemassflag==2
   mssat=mssat-extramass;
end u=input('Please press <Enter> to continue:'); home;

u=('Input total system inertia from Autocad'); disp(u);
vol=input('Enter the volume of the total system from Autocad: ');
a=input('Enter the X moment of inertia:  '); b=input('Enter the Y
moment of inertia:  '); c=input('Enter the Z moment of inertia:
'); d=input('Enter the XY product of inertia value:  ');
e=input('Enter the YZ product of inertia value:  ');
f=input('Enter the XZ product of inertia value:  ');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Below, we multiply the AutoCAD products of inertia by -1 because     %
%the AutoCAD calculation for product of inertia is off by a -1 factor %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

IAcad =[a -d -f;-d b -e;-f -e c]; u=input('Press <enter> to
continue:');

home;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% AutoCAD assumes SIMSAT density equals 1 g/cm^3.  Here we account for the   %
% actual average density of SIMSAT and convert the AutoCAD inertia matrix to %
% kg-m^2 units.                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

density = (mssat*1000)/vol; Iautocad=(IAcad*density)/(1000*100^2);

u='This is the inertia matrix of SIMSAT from the AutoCAD drawing
(in kg m^2 units):'; disp(u); disp(' '); disp(Iautocad);

u=input('Press <enter> to continue.'); home;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Section 5 -- Total SIMSAT mass, SIMSAT composite inertia matrix, and first   %
%              portion of EOM are calculated                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if paymassflag==2 & finemassflag==2
   mssat = mssat + paymass + extramass;
   u= 'The total weight of SIMSAT (in kg) in this configuration is: ';
    disp(u);
   disp(' ');
   disp(mssat);
    Icomp=Iautocad+IpayloadwtC+IextrawtC;
    u= 'The composite SIMSAT inertia matrix, Icomp (in kg*m^2), for the Simulink or MATLAB motion
        simulation is:';
   disp(u);
   disp(' ');
    disp(Icomp);
end

if paymassflag==2 & finemassflag==1
   mssat = mssat + paymass;
   u= 'The total weight of SIMSAT (in kg) in this configuration is: ';
    disp(u);
   disp(' ');
   disp(mssat);
   disp(' ');
```

```
    Icomp=Iautocad+IpayloadwtC;
   u= 'The composite SIMSAT inertia matrix, Icomp (in kg*m^2), for the Simulink or MATLAB motion
       simulation is:';
   disp(u);
   disp(' ');
    disp(Icomp);
end

if paymassflag==1 & finemassflag==2
   mssat = mssat + extramass;
   u= 'The total weight of SIMSAT (in kg) in this configuration is: ';
    disp(u);
   disp(' ');
   disp(mssat);
   disp(' ');
    Icomp=Iautocad+IextrawtC;
    u= 'The composite SIMSAT inertia matrix, Icomp (in kg*m^2), for the Simulink or MATLAB motion
        simulation is:';
   disp(u);
   disp(' ');
    disp(Icomp);
end

if paymassflag==1 & finemassflag==1
   mssat = mssat;
   u= 'The total weight of SIMSAT (in kg) in this configuration is: ';
    disp(u);
   disp(' ');
   disp(mssat);
   disp(' ');
    Icomp=Iautocad;
    u= 'The composite SIMSAT inertia matrix, Icomp (in kg*m^2), for the Simulink or MATLAB motion
        simulation is:';
   disp(u);
   disp(' ');
    disp(Icomp);
end

disp(' '); u='The position vectors (in cm) of the momentum wheels
(FROM the composite center of mass TO the wheel '; disp(u);
u='centers of mass) are shown below.  These position vectors are
displayed for user reference. '; disp(u); u='Should the user ever
wish to run "quikiner.m" , these wheel position vectors will be
```

```
requested as '; disp(u); u='inputs from the user. The user may
wish to print the contents of this page so the data is available
'; disp(u); u='for future reference.  '; disp(u); disp(' ');

rvectorwheel1cm=rvectorwheel1*100;
rvectorwheel2cm=rvectorwheel2*100;
rvectorwheel3cm=rvectorwheel3*100;

u='The position vectors (in cm) of wheel #1, wheel #2 and wheel
#3, respectively, are: '; disp(u); disp(' ');
disp(rvectorwheel1cm); disp(' '); disp(rvectorwheel2cm); disp('
'); disp(rvectorwheel3cm); disp(' ');

%Calculate first portion of the Equations Of Motion
Tempmatrix=-Icomp-Jw1cb-Jw2cb-Jw3cb;
INVTempmatrix=inv(Tempmatrix);
```

A.2 quikiner.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% "quikiner.m" allows the user to directly enter the composite SIMSAT inertia tensor   %
% in kg*m^2 units. This program assumes SIMSAT is already BALANCED (i.e., composite    %
% center of mass very close to [0;0;0] in the body-fixed "b" frame) and assumes the    %
% user knows the composite inertia matrix about the SIMSAT center of mass and the      %
% position vector (in cm) FROM the center of the sphere TO the center of mass for each %
% momentum wheel.  After the user enters the composite inertia tensor and wheel        %
% position vectors, this program calculates the inertia of each momentum wheel (using  %
% the 8 & 5/8" steel hoop and 1/4" aluminum disk baseline design) because the wheel    %
% inertias are used by the SIMULINK and MATLAB motion simulations.  "quikiner.m"       %
% concludes by calculating the inverse matrix of the first four terms (these terms     %
% do not vary with time) of the equations of motion (eom).                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Reference Frames:                                                                 %
% 1) Inertial Reference Frame--x axis points to the north laboratory wall (or other %
%    convenient wall), y axis points at lab ceiling, z axis is deduced from         %
%    right-handed orthogonality.                                                    %
% 2) SIMSAT Body-Fixed Frame (the "b" basis set with origin at center of sphere)    %
% 3) Each momentum wheel has its own body-fixed reference frame (d, f, and h bases) %
%                                                                                   %
% Bases:                                                                           %
% 1) "b basis" (body-fixed basis, origin is at center of central sphere)           %
%    b1 points to the right of this screen                                         %
%    b2 points to the top of this screen                                           %
%    b3 points out of the screen                                                   %
% 2) "d, f, and h bases"--centered on momentum wheels 1, 2 and 3, respectively,    %
%    centers of mass (C.O.M.). If momentum wheel is lying flat on a table, d2, f2  %
%    and h2 point up (parallel with wheel axle hole). d1/f1/h1 and d3/f3/h3 are    %
%    right-handed orthogonal with d2/f2/h2.  The exact pointing directions of      %
%    d1/f1/h1 and d3/f3/h3 are not important as long as right-handed orthogonality %
%    with d2/f2/h2 is satisfied.                                                   %
%                                                                                  %
% Assumptions:                                                                     %
% 1) SIMSAT's center of mass is the physical center of the system ([0;0;0] in the  %
%    SIMSAT "b" frame)                                                             %
%                                                                                  %
% Sections within the code:                                                        %
% Section 1 -- User inputs the SIMSAT composite inertia matrix and wheel position  %
%              vectors                                                             %
% Section 2 -- Calculates inertia matrices of the momentum wheels with respect to  %
%              their own centers of mass and bases.  This section then calculates  %
%              the wheel inertias about their centers of mass with respect to the  %
%              "b" basis.  Finally, this section calculates the inertia matrix of  %
%              each wheel with respect to the "b" basis about the SIMSAT center    %
%              of mass.                                                            %
% Section 3 -- Calculates the inverse matrix of the first portion of the           %
%              equations of motion                                                 %
%                                                                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                                  %
% Definitions of Terms                                                             %
%                                                                                  %
% Inertia Matrices (in kg*m^2):                                                    %
%  Icomp - composite SIMSAT inertia matrix about the SIMSAT center of mass         %
%                                                                                  %
%  Jwheel1 - Inertia of wheel 1 about the wheel 1 center of mass w.r.t. d basis    %
%  Jwheel2 - Inertia of wheel 2 about the wheel 2 center of mass w.r.t. f basis    %
%  Jwheel3 - Inertia of wheel 3 about the wheel 3 center of mass w.r.t. h basis    %
%                                                                                  %
%  Jw1w1b  - Inertia of wheel 1 about the wheel 1 center of mass w.r.t. b basis     %
%  Jw2w2b  - Inertia of wheel 2 about the wheel 2 center of mass w.r.t. b basis     %
%  Jw3w3b  - Inertia of wheel 3 about the wheel 3 center of mass w.r.t. b basis     %
%                                                                                   %
%  Jw1cb - Wheel 1 inertia matrix about the SIMSAT center of mass wrt b basis       %
%  Jw2cb - Wheel 2 inertia matrix about the SIMSAT center of mass wrt b basis       %
%  Jw3cb - Wheel 3 inertia matrix about the SIMSAT center of mass wrt b basis       %
%                                                                                   %
% Transformation Matrices:                                                          %
%  rotaw1 - Transformation Matrix for moving from the d basis to the b basis        %
%  rotaw2 - Transformation Matrix for moving from the f basis to the b basis        %
%  rotaw3 - Transformation Matrix for moving from the h basis to the b basis        %
%  rotbw1 - Transformation Matrix for moving from the b basis to the d basis        %
%  rotbw2 - Transformation Matrix for moving from the b basis to the f basis        %
%  rotbw3 - Transformation Matrix for moving from the b basis to the h basis        %
%                                                                                   %
% Position Vectors:                                                                 %
%  rcomp   - Position vector (written in the "b" basis) FROM the origin at center   %
%            of central sphere TO the SIMSAT composite center of mass.  Since this  %
%            program assumes SIMSAT is balanced, rcomp = [0;0;0].                   %
%  rmw1vcm - Position vector FROM center of sphere TO center of mass of momentum    %
%            wheel #1 in cm (written in the "b" frame)                              %
%  rmw2vcm - Position vector FROM center of sphere TO center of mass of momentum    %
%            wheel #2 in cm (written in the "b" frame)                              %
%  rmw3vcm - Position vector FROM center of sphere TO center of mass of momentum    %
%            wheel #3 in cm (written in the "b" frame)                              %
%                                                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear; home;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Section 1 -- User inputs SIMSAT composite inertia tensor in kg*m^2 units and %
% wheel position vectors in cm                                                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

u='"quikiner.m" allows the user to directly enter the composite
SIMSAT inertia tensor '; disp(u); u='in kg*m^2 units. This program
assumes SIMSAT is already BALANCED (i.e., composite  '; disp(u);
u='center of mass very close to [0;0;0] in the body-fixed "b"
frame) and assumes the  '; disp(u); u='user knows the composite
inertia matrix about the SIMSAT center of mass and the  ';
```

```
disp(u); u='position vector (in cm) FROM the center of the sphere
TO the center of mass for each '; disp(u); u='momentum wheel.
After the user enters the composite inertia tensor and wheel ';
disp(u); u='position vectors, this program calculates the inertia
of each momentum wheel (using '; disp(u); u='the 8 & 5/8" steel
hoop and 1/4" aluminum disk baseline design) because the wheel ';
disp(u); u='inertias are used by the SIMULINK and MATLAB motion
simulations.  "quikiner.m" concludes '; disp(u); u='by calculating
the inverse matrix of the first four terms (these terms do not
vary with time) '; disp(u); u='of the equations of motion (eom).
'; disp(u);

disp(' '); disp(' '); u='Enter the SIMSAT composite inertia
matrix, Icomp, in kg*m^2 units: '; disp(u); disp(' ');

Icomp(1,1)=input('Icomp(1,1) = '); Icomp(1,2)=input('Icomp(1,2) =
'); Icomp(1,3)=input('Icomp(1,3) = ');
Icomp(2,1)=input('Icomp(2,1) = '); Icomp(2,2)=input('Icomp(2,2) =
'); Icomp(2,3)=input('Icomp(2,3) = ');
Icomp(3,1)=input('Icomp(3,1) = '); Icomp(3,2)=input('Icomp(3,2) =
'); Icomp(3,3)=input('Icomp(3,3) = ');

disp(' '); u='Icomp = '; disp(u); disp(' '); disp(Icomp); disp('
'); u=input('Press <enter> to continue:');

%%%%Wheel 1 position vector%%%%
home; disp(' '); u='Enter the position vector (in cm) FROM the
center of the sphere TO the center of mass of momentum wheel #1';
disp(u); disp(' ');

rmw1vcm(1,1)=input('b1 component = '); rmw1vcm(2,1)=input('b2
component = '); rmw1vcm(3,1)=input('b3 component = ');

disp(' '); u='The wheel #1 position vector in cm is :'; disp(u);
disp(' '); disp(rmw1vcm); disp(' '); u=input('Press <enter> to
continue:');

%Convert to meters
rmw1v(1,1)=rmw1vcm(1,1)/100; rmw1v(2,1)=rmw1vcm(2,1)/100;
rmw1v(3,1)=rmw1vcm(3,1)/100;

%%%%Wheel 2 position vector%%%%
home; disp(' '); u='Enter the position vector (in cm) FROM the
```

```
center of the sphere TO the center of mass of momentum wheel #2';
disp(u); disp(' ');

rmw2vcm(1,1)=input('b1 component = '); rmw2vcm(2,1)=input('b2
component = '); rmw2vcm(3,1)=input('b3 component = ');

disp(' '); u='The wheel #2 position vector in cm is :'; disp(u);
disp(' '); disp(rmw2vcm); disp(' '); u=input('Press <enter> to
continue:');

%Convert to meters
rmw2v(1,1)=rmw2vcm(1,1)/100; rmw2v(2,1)=rmw2vcm(2,1)/100;
rmw2v(3,1)=rmw2vcm(3,1)/100;

%%%%Wheel 3 position vector%%%%
home; disp(' '); u='Enter the position vector (in cm) FROM the
center of the sphere TO the center of mass of momentum wheel #3';
disp(u); disp(' ');

rmw3vcm(1,1)=input('b1 component = '); rmw3vcm(2,1)=input('b2
component = '); rmw3vcm(3,1)=input('b3 component = ');

disp(' '); u='The wheel #3 position vector in cm is :'; disp(u);
disp(' '); disp(rmw3vcm); disp(' '); u=input('Press <enter> to
continue:');

%Convert to meters
rmw3v(1,1)=rmw3vcm(1,1)/100; rmw3v(2,1)=rmw3vcm(2,1)/100;
rmw3v(3,1)=rmw3vcm(3,1)/100;



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Section 2 -- Calculates inertia matrices of the momentum wheels with respect to     %
%              their own centers of mass and bases.  This section then calculates     %
%              the wheel inertias about their centers of mass with respect to the     %
%              "b" basis.  Finally, this section calculates the inertia matrix of     %
%              each wheel with respect to the "b" basis about the SIMSAT center       %
%              of mass.                                                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

  rcomp=[0;0;0];
  densityhoop=7.85; %density of steel hoop in g/cc
  densitydisk=2.8;  %density of aluminum disk in g/cc


  r1o=10.95375; %outer radius of steel hoop in cm
  wrim1=.9525;  %width of the steel hoop rim in cm
  t1=3.0111;    %thickness (height) of the steel hoop rim in cm
  tdisk1=.635;  %thickness of the aluminum disk in cm

  r1i=r1o-wrim1;
  m1o=(densityhoop*pi*r1o^2*t1)/1000;
  m1i=(densityhoop*pi*r1i^2*t1)/1000;
  mhoop1=m1o-m1i;
  mdisk1=(densitydisk*pi*r1i^2*tdisk1)/1000;
  m1=mhoop1+mdisk1; %total mass of momentum wheel 1 in kg

  m2o=m1o;
  m3o=m1o;
  m2i=m1i;
  m3i=m1i;
  r2o=r1o;
  r3o=r1o;
  r2i=r1i;
  r3i=r1i;
  t2=t1;
  t3=t1;
  tdisk2=tdisk1;
  tdisk3=tdisk1;
  m2=m1;
  m3=m1;
  mhoop2=mhoop1;
  mhoop3=mhoop1;
  mdisk2=mdisk1;
  mdisk3=mdisk1;

%Convert r and t to meters
  r1o=r1o/100;
  r1i=r1i/100;
  t1=t1/100;
  tdisk1=tdisk1/100;
  r2o=r2o/100;
  r2i=r2i/100;
```

```
 t2=t2/100;
 tdisk2=tdisk2/100;
 r3o=r3o/100;
 r3i=r3i/100;
 t3=t3/100;
 tdisk3=tdisk3/100;

%Calculate inertia matrix for thin aluminum disk
 J11=(3*r1i^2+tdisk1^2)/12;
 J12=(r1i^2)/2;
 J13=(3*r1i^2+tdisk1^2)/12;
 Jdisk1=mdisk1*[J11 0 0; 0 J12 0; 0 0 J13];

 J21=(3*r2i^2+tdisk2^2)/12;
 J22=(r2i^2)/2;
 J23=(3*r2i^2+tdisk2^2)/12;
 Jdisk2=mdisk2*[J21 0 0; 0 J22 0; 0 0 J23];

 J31=(3*r3i^2+tdisk3^2)/12;
 J32=(r3i^2)/2;
 J33=(3*r3i^2+tdisk3^2)/12;
 Jdisk3=mdisk3*[J31 0 0; 0 J32 0; 0 0 J33];

 %Calculating inertia matrix for outer radius of steel hoop
 Jo11=(3*r1o^2+t1^2)/12;
 Jo12=(r1o^2)/2;
 Jo13=(3*r1o^2+t1^2)/12;
 Johoop1=[Jo11 0 0; 0 Jo12 0; 0 0 Jo13];

 Jo21=(3*r2o^2+t2^2)/12;
 Jo22=(r2o^2)/2;
 Jo23=(3*r2o^2+t2^2)/12;
 Johoop2=[Jo21 0 0; 0 Jo22 0; 0 0 Jo23];

 Jo31=(3*r3o^2+t3^2)/12;
 Jo32=(r3o^2)/2;
 Jo33=(3*r3o^2+t3^2)/12;
 Johoop3=[Jo31 0 0; 0 Jo32 0; 0 0 Jo33];

 %Calculating inertia matrix for inner radius of steel hoop
 Ji11=(3*r1i^2+t1^2)/12;
 Ji12=(r1i^2)/2;
 Ji13=(3*r1i^2+t1^2)/12;
```

```
  Jihoop1=[Ji11 0 0; 0 Ji12 0; 0 0 Ji13];

  Ji21=(3*r2i^2+t2^2)/12;
  Ji22=(r2i^2)/2;
  Ji23=(3*r2i^2+t2^2)/12;
  Jihoop2=[Ji21 0 0; 0 Ji22 0; 0 0 Ji23];

  Ji31=(3*r3i^2+t3^2)/12;
  Ji32=(r3i^2)/2;
  Ji33=(3*r3i^2+t3^2)/12;
  Jihoop3=[Ji31 0 0; 0 Ji32 0; 0 0 Ji33];

%Calculating inertia matrix for the steel hoop
  Jhoop1 = m1o*Johoop1 - m1i*Jihoop1;
  Jhoop2 = m2o*Johoop2 - m2i*Jihoop2;
  Jhoop3 = m3o*Johoop3 - m3i*Jihoop3;

%Calculate inertia matrix for entire wheel
  Jwheel1=Jdisk1+Jhoop1;
  Jwheel2=Jdisk2+Jhoop2;
  Jwheel3=Jdisk3+Jhoop3;

%Note:  rotaw1 is wheel 1 rotation matrix going one way, 1-2-3
%User will input angles for rotation matrices to orient momentum wheels

%u=['Initial orientation of all momentum wheels is with their second axis (axis of rotation)'];
%disp(u);
%u=['pointing along respective Simsat "b" axis.  The following rotations will align wheel axes with
%    Simsat axes:'];
%disp(u);

%Momentum Wheel 1 %
thetarot1w1=0;%input('How many degrees do you want to rotate wheel 1 about its 1st axis? ');
thetarot2w1=0;%input('How many degrees do you want to rotate wheel 1 about its new 2nd axis? ');
thetarot3w1=90;%input('How many degrees do you want to rotate wheel 1 about its new 3rd axis? ');

thetarot1w1rad=thetarot1w1*pi/180;
thetarot2w1rad=thetarot2w1*pi/180;
thetarot3w1rad=thetarot3w1*pi/180;

rot1w1=[1 0 0;0 cos(thetarot1w1rad) sin(thetarot1w1rad);0
-sin(thetarot1w1rad) cos(thetarot1w1rad)];
rot2w1=[cos(thetarot2w1rad) 0 -sin(thetarot2w1rad);0 1
```

```
0;sin(thetarot2w1rad) 0 cos(thetarot2w1rad)];
rot3w1=[cos(thetarot3w1rad) sin(thetarot3w1rad)
0;-sin(thetarot3w1rad) cos(thetarot3w1rad) 0;0 0 1];

rotaw1=rot3w1*(rot2w1*rot1w1); rotbw1=rotaw1'; Jw1w1d=Jwheel1;
Jw1w1b=rotaw1*(Jw1w1d*rotbw1);

%Momentum Wheel 2 %
thetarot1w2=0;%input('Moving on to wheel 2.  How many degrees do you want to rotate wheel 2 about
              %its 1st axis? ');
thetarot2w2=0;%input('How many degrees do you want to rotate wheel 2 about its new 2nd axis?');
thetarot3w2=0;%input('How many degrees do you want to rotate wheel 2 about its new 3rd axis?');

thetarot1w2rad=thetarot1w2*pi/180;
thetarot2w2rad=thetarot2w2*pi/180;
thetarot3w2rad=thetarot3w2*pi/180;

rot1w2=[1 0 0;0 cos(thetarot1w2rad) sin(thetarot1w2rad);0
-sin(thetarot1w2rad) cos(thetarot1w2rad)];
rot2w2=[cos(thetarot2w2rad) 0 -sin(thetarot2w2rad);0 1
0;sin(thetarot2w2rad) 0 cos(thetarot2w2rad)];
rot3w2=[cos(thetarot3w2rad) sin(thetarot3w2rad)
0;-sin(thetarot3w2rad) cos(thetarot3w2rad) 0;0 0 1];

rotaw2=rot3w2*(rot2w2*rot1w2); rotbw2=rotaw2'; Jw2w2f=Jwheel2;
Jw2w2b=rotaw2*(Jw2w2f*rotbw2);


%Momentum Wheel 3 %
thetarot1w3=-90;%input('Finally, enter data for wheel 3.  How many degrees do you want to rotate
                %wheel 3 about its 1st axis?');
thetarot2w3=0;%input('How many degrees do you want to rotate wheel 3 about its new 2nd axis?');
thetarot3w3=0;%input('How many degrees do you want to rotate wheel 3 about its new 3rd axis?');

thetarot1w3rad=thetarot1w3*pi/180;
thetarot2w3rad=thetarot2w3*pi/180;
thetarot3w3rad=thetarot3w3*pi/180;

rot1w3=[1 0 0;0 cos(thetarot1w3rad) sin(thetarot1w3rad);0
-sin(thetarot1w3rad) cos(thetarot1w3rad)];
rot2w3=[cos(thetarot2w3rad) 0 -sin(thetarot2w3rad);0 1
0;sin(thetarot2w3rad) 0 cos(thetarot2w3rad)];
rot3w3=[cos(thetarot3w3rad) sin(thetarot3w3rad)
```

```
0;-sin(thetarot3w3rad) cos(thetarot3w3rad) 0;0 0 1];

rotaw3=rot3w3*(rot2w3*rot1w3); rotbw3=rotaw3'; Jw3w3h=Jwheel3;
Jw3w3b=rotaw3*(Jw3w3h*rotbw3);

%u = 'Momentum Wheel Inertia Matrices (Momentum wheel mass center)';
%disp(u);
Jw1w1b; Jw2w2b; Jw3w3b;


% Momentum Wheel Inertia Matrices about the SIMSAT composite center of mass %
rvectorwheel1=rmw1v-rcomp; rvectorwheel2=rmw2v-rcomp;
rvectorwheel3=rmw3v-rcomp;

paraxis1=[rvectorwheel1(2)^2+rvectorwheel1(3)^2
-rvectorwheel1(1)*rvectorwheel1(2)
-rvectorwheel1(1)*rvectorwheel1(3);-rvectorwheel1(1)*rvectorwheel1(2)
rvectorwheel1(1)^2+rvectorwheel1(3)^2
-rvectorwheel1(2)*rvectorwheel1(3);-rvectorwheel1(1)*rvectorwheel1(3)
-rvectorwheel1(2)*rvectorwheel1(3)
rvectorwheel1(1)^2+rvectorwheel1(2)^2]; Jw1cb=Jw1w1b+m1*paraxis1;

paraxis2=[rvectorwheel2(2)^2+rvectorwheel2(3)^2
-rvectorwheel2(1)*rvectorwheel2(2)
-rvectorwheel2(1)*rvectorwheel2(3);-rvectorwheel2(1)*rvectorwheel2(2)
rvectorwheel2(1)^2+rvectorwheel2(3)^2
-rvectorwheel2(2)*rvectorwheel2(3);-rvectorwheel2(1)*rvectorwheel2(3)
-rvectorwheel2(2)*rvectorwheel2(3)
rvectorwheel2(1)^2+rvectorwheel2(2)^2]; Jw2cb=Jw2w2b+m2*paraxis2;

paraxis3=[rvectorwheel3(2)^2+rvectorwheel3(3)^2
-rvectorwheel3(1)*rvectorwheel3(2)
-rvectorwheel3(1)*rvectorwheel3(3);-rvectorwheel3(1)*rvectorwheel3(2)
rvectorwheel3(1)^2+rvectorwheel3(3)^2
-rvectorwheel3(2)*rvectorwheel3(3);-rvectorwheel3(1)*rvectorwheel3(3)
-rvectorwheel3(2)*rvectorwheel3(3)
rvectorwheel3(1)^2+rvectorwheel3(2)^2]; Jw3cb=Jw3w3b+m3*paraxis3;



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Section 3 -- Calculates the inverse matrix of the first portion of the    %
%              equations of motion                                          %
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Tempmatrix=-Icomp-Jw1cb-Jw2cb-Jw3cb;
INVTempmatrix=inv(Tempmatrix);

disp(' '); disp(' '); u='This program is complete.  You can now
run motion simulations (such as "simcloop3.m") in SIMULINK or
MATLAB. '; disp(u); disp(' ');
```

A.3 simcloop3.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% "simcloop3.m"--Closed Loop Simulation with ODE45 solving for entire system state %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%% Initialize control inputs thetades, ratedes, and wheelrpmdes %%%%%%%%%%%%%%%%%%%%
rolldes=0; yawdes=0; pitchdes=0;

rollratedes=0; yawratedes=0; pitchratedes=0;

wheel1rpmdes=0; wheel2rpmdes=0; wheel3rpmdes=0;

%%%%%%% Initialize K4, proportional rate gain used for options 2, 3 and 4 %%%%%%%%%%%%%%%%%%%
k4=[1 0 0;0 1 0;0 0 1];

%%%%%%%%%%%%% User input section %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp('Choose one of the following options: ') disp('Enter [1] for
Target mode (user specifies desired Euler angles for Simsat)')
disp('Enter [2] for Target mode with rollrate') disp('Enter [3]
for Roll Spin mode') disp('Enter [4] for Yaw Spin mode')
disp('Enter [5] for Wheel RPM mode (user specifies desired
momentum wheel speeds)') disp('
') choice=input('Enter option number here '); disp('
')

if choice==1

    rolldes=input('Enter desired roll angle (range is -180 to +180 degrees) ');
    while abs(rolldes)>180
         disp('Your input was outside specified range');
         rolldes=input('Enter desired roll angle (range is -180 to +180 degrees) ');
      end
    yawdes=input('Enter desired yaw angle (range is -360 to +360 degrees) ');
      while abs(yawdes)>360
         disp('Your input was outside specified range');
         yawdes=input('Enter desired yaw angle (range is -360 to 360 degrees) ');
      end
    pitchdes=input('Enter desired pitch angle (range is -25 to +25 degrees) ');
        while abs(pitchdes)>25
         disp('Your input was outside specified range');
         pitchdes=input('Enter desired pitch angle (range is -25 to +25 degrees) ');
      end

   graphflag='Option 1 (Target Mode)';
end

if choice==2

    yawdes=input('Enter desired yaw angle (range is -360 to +360 degrees) ');
      while (yawdes)>360
         disp('Your input was outside specified range');
         yawdes=input('Enter desired yaw angle (range is -360 to +360 degrees) ');
      end
    pitchdes=input('Enter desired pitch angle (range is -25 to +25 degrees) ');
        while abs(pitchdes)>25
         disp('Your input was outside specified range');
         pitchdes=input('Enter desired pitch angle (range is -25 to +25 degrees) ');
      end
   rollratedes=input('Enter desired roll rate (range is -12 to +12 RPM) ');
    while abs(rollratedes)>12
        disp('Your input was outside specified range');
         rollratedes=input('Enter desired roll rate (range is -12 to +12 RPM) ');
      end

   %%% Display RPM to deg/sec conversion on screen for user %%%%%%%%%%%%%%%%
   degrollrate=rollratedes*(2*pi/60)*(180/pi);
   show=[num2str(rollratedes),' RPM = ',num2str(degrollrate),' deg/sec'];
   disp(show);

   %%%%%%%%% Input K4 proportional rollrate gain %%%%%%%%%%%%%%%%%%%%%%%%%%
   k4rollrate=input('K4 proportional rollrate gain = ');
   k4=[k4rollrate 0 0;0 1 0;0 0 1];
   graphflag='Option 2 (Target Mode with Roll Rate)';
```

```
end

if choice==3
   rollratedes=input('Enter desired roll rate (range is -12 to +12 RPM) ');
    while abs(rollratedes)>12
        disp('Your input was outside specified range');
        rollratedes=input('Enter desired roll rate (range is -12 to +12 RPM) ');
      end

   %%% Display RPM to deg/sec conversion on screen for user %%%%%%%%%%%%%%%%
   degrollrate=rollratedes*(2*pi/60)*(180/pi);
   show=[num2str(rollratedes),' RPM = ',num2str(degrollrate),' deg/sec'];
   disp(show);

   %%%%%%%%% Input K4 proportional rollrate gain %%%%%%%%%%%%%%%%%%%%%%%%%
   k4rollrate=input('K4 proportional rollrate gain = ');
    k4=[k4rollrate 0 0;0 1 0;0 0 1];

   graphflag='Option 3 (Roll Spin Mode)';
end

if choice==4
    yawratedes=input('Enter desired yaw rate (range is -2.6 to +2.6 RPM) ');
    while abs(yawratedes)>2.6
        disp('Your input was outside specified range');
        yawratedes=input('Enter desired yaw rate (range is -2.6 to +2.6 RPM) ');
      end

   %%% Display RPM to deg/sec conversion on screen for user %%%%%%%%%%%%%%%%
   degyawrate=yawratedes*(2*pi/60)*(180/pi);
   show=[num2str(yawratedes),' RPM = ',num2str(degyawrate),' deg/sec'];
   disp(show);

   %%%%%%%%% Input K4 proportional yawrate gain %%%%%%%%%%%%%%%%%%%%%%%%%
   k4yawrate=input('K4 proportional yawrate gain = ');
   k4=[1 0 0;0 k4yawrate 0;0 0 1];

   graphflag='Option 4 (Yaw Spin Mode)';
end

if choice==5
   wheel1rpmdes=input('Enter desired speed for wheel 1 (range is -200 to +200 RPM) ');
    while abs(wheel1rpmdes)>200
```

```
        disp('Your input was outside specified range');
            wheel1rpmdes=input('Enter desired speed for wheel 1 (range is -200 to +200 RPM) ');
        end
    wheel2rpmdes=input('Enter desired speed for wheel 2 (range is -200 to +200 RPM) ');
    while abs(wheel2rpmdes)>200
        disp('Your input was outside specified range');
            wheel2rpmdes=input('Enter desired speed for wheel 2 (range is -200 to +200 RPM) ');
        end

   wheel3rpmdes=input('Enter desired speed for wheel 3 (range is -200 to +200 RPM) ');
    while abs(wheel3rpmdes)>200
        disp('Your input was outside specified range');
            wheel3rpmdes=input('Enter desired speed for wheel 3 (range is -200 to +200 RPM) ');
        end

   %%% Display RPM to rad/sec conversion on screen for user %%%%%%%%%%%%%%%%
   disp('                                      ');
   radwheel1=wheel1rpmdes*(2*pi/60);
   show1=[num2str(wheel1rpmdes),' RPM = ',num2str(radwheel1),' rad/sec'];
   disp(show1);

   radwheel2=wheel2rpmdes*(2*pi/60);
   show2=[num2str(wheel2rpmdes),' RPM = ',num2str(radwheel2),' rad/sec'];
   disp(show2);

   radwheel3=wheel3rpmdes*(2*pi/60);
   show3=[num2str(wheel3rpmdes),' RPM = ',num2str(radwheel3),' rad/sec'];
   disp(show3);

   graphflag='Option 5 (Wheel RPM Mode)';
end

%%%%%%%%%%%%%%%%% Assemble control input vectors %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
thetades=[rolldes;yawdes;pitchdes];
ratedes=[rollratedes;yawratedes;pitchratedes];
wheelrpmdes=[wheel1rpmdes;wheel2rpmdes;wheel3rpmdes];

%%%%%%%%%%%% Convert user input units to radians/MKS unit system  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
thetades=thetades*(pi/180); %%convert from degrees to radians
ratedes=ratedes*(2*pi/60); %%convert from RPM to rad/sec
wheelrpmdes=wheelrpmdes*(2*pi/60); %%convert from RPM to rad/sec

%%%%%%%%%%%%%%% Control gain input section %%%%%%%%%%%%%%%%%%%%%%%%%
```

```
disp('
') disp('Enter appropriate theta gains, K1, and rate gains, K2,
for your selected mode option ') disp('
')


k1roll  =input('k1roll= '); k1yaw   =input('k1yaw= '); k1pitch
=input('k1pitch= ');

k2rollrate  =input('k2rollrate= '); k2yawrate   =input('k2yawrate=
'); k2pitchrate =input('k2pitchrate= ');

k1=[k1roll 0 0;0 k1yaw 0;0 0 k1pitch]; k2=[k2rollrate 0 0;0
k2yawrate 0;0 0 k2pitchrate]; k3=.0585

Ctorqw1max=188;     %in oz-in
Ctorqw2max=188;     %in oz-in
Ctorqw3max=188;     %in oz-in

%%%%%%%%%%%%%%% Input simulation time %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('                                               ')
disp('Enter simulation end time (in seconds): ')
finaltime=input('final time= '); disp('Enter maneuver number (for
graphing purposes): '); maneuvernumber=input('number = ');

%%%%%%%%%%%%%%%% Start Simulation  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                               %
% Definitions of the omega vector                                               %
%                                                                               %
% omega(1:3)= Momentum wheel #1 speed in rad/sec (1st & 3rd component are zero) %
% omega(4:6)= Momentum wheel #2 speed in rad/sec (1st & 3rd component are zero) %
% omega(7:9)= Momentum wheel #3 speed in rad/sec (1st & 3rd component are zero) %
% omega(10) = omega1 of Simsat in rad/sec                                       %
% omega(11) = omega2 of Simsat in rad/sec                                       %
% omega(12) = omega3 of Simsat in rad/sec                                       %
% omega(13) = Roll angle in rad                                                 %
% omega(14) = Yaw angle in rad                                                  %
% omega(15) = Pitch angle in rad                                                %
%                                                                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

tspan1=[0:0.1:finaltime]; flag='';
initialomega=[0;0;0;0;0;0;0;0;0;0;0;0;0;0;0];
[t,omega]=ode45('domega2',tspan1,initialomega,[],thetades,ratedes,wheelrpmdes,k1,k2,k3,k4,...
          Ctorqw1max,Ctorqw2max,Ctorqw3max,rotaw1,rotaw2,rotaw3,Icomp,Jw1w1d,Jw2w2f,Jw3w3h,...
          Jw1cb,Jw2cb,Jw3cb,INVTempmatrix);

%%%%%%%%%% Assemble vectors for plotting purposes %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
plotwheel1speed=omega(:,2); plotwheel2speed=omega(:,5);
plotwheel3speed=omega(:,8);

plotrollratemeas=omega(:,10)*180/pi;
plotyawratemeas=omega(:,11)*180/pi;
plotpitchratemeas=omega(:,12)*180/pi;

plotrollanglemeas=omega(:,13)*180/pi;
plotyawanglemeas=omega(:,14)*180/pi;
plotpitchanglemeas=omega(:,15)*180/pi;

%%%%%%%%%%%%% Plot Momentum wheel speeds vs time %%%%%%%%%%%%%%%%%%
figure(1)
subplot(2,2,3),plot(t,plotwheel1speed,'b-',t,plotwheel2speed,'g--',t,plotwheel3speed,'r:')
title('Momentum wheel speeds: Wheel1(solidblue),
Wheel2(dashedgreen), Wheel3(dottedred)') xlabel('Time (sec)')
ylabel('Wheel speed (rad/sec)')

%%%%%%%%%%%%%% Plot Simsat angular velocities %%%%%%%%%%%%%%%%%%%%%%%
subplot(2,2,2),plot(t,plotrollratemeas,'b-',t,plotyawratemeas,'g--',t,plotpitchratemeas,'r:')
title('SIMSAT rollrate(solidblue), yawrate(dashedgreen),
pitchrate(dottedred)') xlabel('Time (sec)') ylabel('SIMSAT angular
velocity (deg/sec)')

%%%%%%%%%%%%%% Plot Simsat euler angles %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
subplot(2,2,1),plot(t,plotrollanglemeas,'b-',t,plotyawanglemeas,'g--',t,plotpitchanglemeas,'r:')
title('Roll(solidblue), Yaw(dashedgreen), and Pitch(dottedred)')
xlabel('Time (sec)') ylabel('SIMSAT attitude angles (deg)')
man=num2str(maneuvernumber); gtext({graphflag,'Maneuver # ',man})
```

A.4 domega2.m

```
%%%%%%%%%%%%%%%
% "domega2.m" %
%%%%%%%%%%%%%%%

function domega2 =
domega2(t,omega,flag,thetades,ratedes,wheelrpmdes,k1,k2,k3,k4,Ctorqw1max,Ctorqw2max,Ctorqw3max,...
rotaw1,rotaw2,rotaw3,Icomp,Jw1w1d,Jw2w2f,Jw3w3h,Jw1cb,Jw2cb,Jw3cb,INVTempmatrix);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
% Assumptions                                               %
% 1) No losses in the motor                                 %
% 2) No disturbance torques                                 %
%                                                           %
% Note!:  Rotor inertia of smartmotor has NOT been included %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                                  %%
% Definitions of Terms                                                              %
%                                                                                   %
% Angular Velocity (in rad/sec)                                                     %
%    ww1d - Angular Velocity of wheel one with respect to the d frame               %
%    ww2f - Angular Velocity of wheel two with respect to the f frame               %
%    ww3h - Angular Velocity of wheel three with respect to the h frame             %
%                                                                                   %
% Angular Acceleration (in rad/sec)                                                 %
% ww1ddot - Angular acceleration of wheel one with respect to the d basis           %
% ww2fdot - Angular acceleration of wheel two with respect to the f basis           %
% ww3hdot - Angular acceleration of wheel three with respect to the h basis         %
%                                                                                   %
% Inertia Matrices (in kg*m^2)                                                      %
%  Jw1w1d - Inertia of wheel one about the wheel one center of mass wrt d basis     %
%  Jw2w2f - Inertia of wheel two about the wheel two center of mass wrt f basis     %
%  Jw3w3h - Inertia of wheel three about the wheel three center of mass wrt h basis %
%                                                                                   %
%    Iscb - Simsat inertia matrix about the Simsat center of mass wrt b basis       %
%   Jw1cb - Wheel one inertia matrix about the Simsat center of mass wrt b basis    %
%   Jw2cb - Wheel two inertia matrix about the Simsat center of mass wrt b basis    %
%   Jw3cb - Wheel three inertia matrix about the Simsat center of mass wrt b basis  %
%                                                                                   %
% Transformation Matrices                                                           %
%  rotaw1 - Transformation Matrix for moving from the d basis to the b basis        %
%  rotaw2 - Transformation Matrix for moving from the f basis to the b basis        %
%  rotaw3 - Transformation Matrix for moving from the h basis to the b basis        %
%                                                                                   %
%                                                                                   %
%                                                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

domega2=zeros(15,1);

thetameas=[omega(13);omega(14);omega(15)];
ratemeas=[omega(10);omega(11);omega(12)];



ww1d=omega(1:3); ww2f=omega(4:6); ww3h=omega(7:9);

wheelspeed=[omega(2);omega(5);omega(8)];

%%%%%%%% Block diagram algebra %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

u=thetades-thetameas; g=k1*u; v=g-k2*ratemeas+ratedes; y=k4*v;
wdes=-y; z=wdes-wheelspeed+wheelrpmdes; Tcom=k3*z;

%%%%%% Torque on MW = minimum of commanded torque vs. max continuous torque capability of motor %%
if Tcom(1,1) >= 0
   Tasfuncwspeed1=.007061*(Ctorqw1max-exp(.0193942*abs(wheelspeed(1,1))));
    %%%%%%%%% Detect wheel 1 saturation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if abs(wheelspeed(1,1))>=270
        Tasfuncwspeed1=0;
        end
    Tcapablew1=min(Tcom(1,1),Tasfuncwspeed1);
end

if Tcom(2,1) >= 0
   Tasfuncwspeed2=.007061*(Ctorqw2max-exp(.0193942*abs(wheelspeed(2,1))));
    %%%%%%%%% Detect wheel 2 saturation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if abs(wheelspeed(2,1))>=270
        Tasfuncwspeed2=0;
        end
    Tcapablew2=min(Tcom(2,1),Tasfuncwspeed2);
```

```
end


if Tcom(3,1) >= 0
    Tasfuncwspeed3=.007061*(Ctorqw3max-exp(.0193942*abs(wheelspeed(3,1))));
   %%%%%%%%% Detect wheel 3 saturation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if abs(wheelspeed(3,1))>=270
        Tasfuncwspeed3=0;
        end
   Tcapablew3=min(Tcom(3,1),Tasfuncwspeed3);
end



if Tcom(1,1) < 0
    negTasfuncwspeed1=-(.007061*(Ctorqw1max-exp(.0193942*abs(wheelspeed(1,1)))));
   %%%%%%%%% Detect wheel 1 saturation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if abs(wheelspeed(1,1))>=270
        negTasfuncwspeed1=0;
        end
   Tcapablew1=-min(abs(Tcom(1,1)),abs(negTasfuncwspeed1));
end


if Tcom(2,1) < 0
    negTasfuncwspeed2=-(.007061*(Ctorqw2max-exp(.0193942*abs(wheelspeed(2,1)))));
    %%%%%%%%% Detect wheel 2 saturation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if abs(wheelspeed(2,1))>=270
        negTasfuncwspeed2=0;
        end
    Tcapablew2=-min(abs(Tcom(2,1)),abs(negTasfuncwspeed2));
end


if Tcom(3,1) < 0
    negTasfuncwspeed3=-(.007061*(Ctorqw3max-exp(.0193942*abs(wheelspeed(3,1)))));
   %%%%%%%%% Detect wheel 3 saturation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if abs(wheelspeed(3,1))>=270
        negTasfuncwspeed3=0;
        end
    Tcapablew3=-min(abs(Tcom(3,1)),abs(negTasfuncwspeed3));
end



T=[Tcapablew1;Tcapablew2;Tcapablew3];


%%%%% Assume all 3 wheels have same moment of inertia %%%%%%%%%%%%%%%%%
```

```
wheelaccel=T/Jw1w1d(2,2);

ww1ddot=[0;wheelaccel(1,1);0]; ww2fdot=[0;wheelaccel(2,1);0];
ww3hdot=[0;wheelaccel(3,1);0];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
% For this section, we are solving for the second part of our system           %
% term by term.  To understand what we are talking about, review the           %
% equations of motion within the ADACS portion of the detailed design chapter. %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

term1=rotaw1*(Jw1w1d*ww1ddot(1:3));

term2=rotaw2*(Jw2w2f*ww2fdot(1:3));

term3=rotaw3*(Jw3w3h*ww3hdot(1:3));

temp4=Icomp*omega(10:12); term4=cross(omega(10:12),temp4);

temp5=Jw1cb*omega(10:12); term5=cross(omega(10:12),temp5);

temp6=rotaw1*(Jw1w1d*ww1d); term6=cross(omega(10:12),temp6);

temp7=Jw2cb*omega(10:12); term7=cross(omega(10:12),temp7);

temp8=rotaw2*(Jw2w2f*ww2f); term8=cross(omega(10:12),temp8);

temp9=Jw3cb*omega(10:12); term9=cross(omega(10:12),temp9);

temp10=rotaw3*(Jw3w3h*ww3h); term10=cross(omega(10:12),temp10);


Secondportion=term1+term2+term3+term4+term5+term6+term7+term8+term9+term10;

%%%%%%%%%%%%%%Calculate equations of motion by finding omegadot  **********************************

domega2(1:3)=ww1ddot; domega2(4:6)=ww2fdot; domega2(7:9)=ww3hdot;
domega2(10:12)=INVTempmatrix*Secondportion;
```

```
%%%%%% Euler angles for roll angle, yaw angle, and pitch angle %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
domega2(13) = omega(10) + omega(11)*-cos(omega(13))*tan(omega(15))
+ omega(12)*sin(omega(13))*tan(omega(15)); domega2(14) =
omega(11)*(cos(omega(13))/cos(omega(15))) -
omega(12)*(sin(omega(13))/cos(omega(15))); domega2(15) =
omega(11)*sin(omega(13)) + omega(12)*cos(omega(13));
```

A.5 solver2play.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% "solver2play.m"--Uses Bang-Bang control input--speed motor up for 5 seconds,      %
%                  slow motor down for 5 seconds                                    %
%                --Uses ODE45 routine to solve EOM                                  %
%                --Assumes 36V applied to motor                                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                  %
% Definitions of the omega vector                                  %
%                                                                  %
% omega(1) = ww1d of momentum wheel 1 in rad/sec                   %
% omega(2) = ww2f of momentum wheel 2 in rad/sec                   %
% omega(3) = ww3h of momentum wheel 3 in rad/sec                   %
% omega(4) = omega1 of Simsat in rad/sec                           %
% omega(5) = omega2 of Simsat in rad/sec                           %
% omega(6) = omega3 of Simsat in rad/sec                           %
% omega(7) = Roll angle                                            %
% omega(8) = Yaw angle                                             %
% omega(9) = Pitch angle                                           %
%                                                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Solve EOM for  first 5 sec %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%I1=0
%I2=4
%I3=0
%V=36

Ctorqw1max=0                         %in oz-in
```

```
Ctorqw2max=188                       %continuous torque in oz-in for 36V case
Ctorqw3max=0                         %in oz-in

%Find motor #2 correction term to prevent infinite torque at zero wheel speed
%Kt=18.5;                              %From smart motor spec sheet in (oz-in/amp)
%Tbattdrawmotor2=I2*Kt;                %Motor 2 torque based on amp draw from battery
%convozin2Nm=.007061;
%C1=I2*V/(Tbattdrawmotor2*convozin2Nm)

tspan1=[0:.01:5]; flag=''; initialomega1=[0;0;0;0;0;0;0;0]
torqdir=1
[t1,omega1]=ode45('domega',tspan1,initialomega1,[],Ctorqw1max,Ctorqw2max,Ctorqw3max,torqdir,...
            rotaw1,rotaw2,rotaw3,Icomp,Jw1w1d,Jw2w2f,Jw3w3h,Jw1cb,Jw2cb,Jw3cb,INVTempmatrix);
TorqueW2a=.007061*(Ctorqw2max-exp(.0193942*omega1(:,2)));

%[t1,omega1]=ode45('domega',tspan1,initialomega1,[],I1,I2,I3,V,C1,rotaw1,rotaw2,rotaw3,Icomp,...
%            Jw1w1d,Jw2w2f,Jw3w3h,Jw1cb,Jw2cb,Jw3cb,INVTempmatrix);
%TorqueW1a=(I1*V)./(omega1(:,1)+C1);
%TorqueW2a=(I2*V)./(omega1(:,2)+C1);
%TorqueW3a=(I3*V)./(omega1(:,3)+C1);
%wwheeldot0to5sec=I2*V./(Jw2w2f(2,2)*(omega1(:,2)+C1));

%Solve EOM for seconds 5-10 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%I1=0
%I2=-I2
%I3=0


tspan2=[5.01:.01:10]; initialomega2=omega1(501,:); torqdir=2
[t2,omega2]=ode45('domega',tspan2,initialomega2,[],Ctorqw1max,Ctorqw2max,Ctorqw3max,torqdir,...
            rotaw1,rotaw2,rotaw3,Icomp,Jw1w1d,Jw2w2f,Jw3w3h,Jw1cb,Jw2cb,Jw3cb,INVTempmatrix);
%[t2,omega2]=ode45('domega',tspan2,initialomega2,[],I1,I2,I3,V,C1,rotaw1,rotaw2,rotaw3,Icomp,...
%            Jw1w1d,Jw2w2f,Jw3w3h,Jw1cb,Jw2cb,Jw3cb,INVTempmatrix);
%TorqueW1b=(I1*V)./(omega2(:,1)+C1);
%TorqueW2b=(I2*V)./(omega2(:,2)+C1);
%TorqueW3b=(I3*V)./(omega2(:,3)+C1);

%Solve EOM for seconds 10-15 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%I1=0;
%I2=0;
%I3=0;
%tspan3=[10.01:.01:15];
%initialomega3=omega2(500,:);
```

```
%[t3,omega3]=ode45('domega',tspan3,initialomega3,[],I1,I2,I3,V,C1,rotaw1,rotaw2,rotaw3,Icomp,...
%            Jw1w1d,Jw2w2f,Jw3w3h,Jw1cb,Jw2cb,Jw3cb,INVTempmatrix);
%TorqueW1c=(I1*V)./(omega3(:,1)+C1);
%TorqueW2c=(I2*V)./(omega3(:,2)+C1);
%TorqueW3c=(I3*V)./(omega3(:,3)+C1);

%Assemble final vectors for plotting and display purposes %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
finalomega=[omega1;omega2]; finaltime=[t1;t2];
%finaltorqueW1=TorqueW1a;              %finaltorqueW1=[TorqueW1a;TorqueW1b;TorqueW1c];
finaltorqueW2=TorqueW2a;               %finaltorqueW2=[TorqueW2a;TorqueW2b;TorqueW2c];
%finaltorqueW3=TorqueW3a;              %finaltorqueW3=[TorqueW3a;TorqueW3b;TorqueW3c];
yawangleat10sec=finalomega(1001,8)
maxwheelspeedat5sec=finalomega(501,2)




%Plot Momentum wheel speeds vs time
figure(1)
subplot(2,2,3),plot(finaltime,finalomega(:,1),'-',finaltime,finalomega(:,2),'--',
                    finaltime,finalomega(:,3),':')
title('Momentum Wheel Speeds:
Wheel1(solid), Wheel2(dashed), Wheel3(dotted)') xlabel('Time
(sec)') ylabel('Wheel speed (rad/sec)') text(5.7,155,'Wheel 2')
text(2.5,18,'Wheels 1 and 3 motionless')

%Plot Torque vs Momentum Wheel speeds
%figure(2)
%plot(omega1(:,1),finaltorqueW1,'-')
%title('Torque vs Wheel 1 speed for 36V')
%xlabel('Wheel 1 speed (rad/sec)')
%ylabel('Torque (N-m/rad)')

%figure(3)
%plot(omega1(:,2),finaltorqueW2,'--')
%title('Torque vs Wheel 2 speed for 36V')
%xlabel('Wheel 2 speed (rad/sec)')
%ylabel('Torque (N-m)')

%figure(4)
%plot(omega1(:,3),finaltorqueW3,':')
%title('Torque vs Wheel 3 speed for 36V')
%xlabel('Wheel 3 speed (rad/sec)')
%ylabel('Torque (N-m/rad)')
```

```
%Plot Simsat angular velocities
%Convert to degrees/second%%%%%%%%%%%%%
finalomega(:,4)=finalomega(:,4)*180/pi;
finalomega(:,5)=finalomega(:,5)*180/pi;
finalomega(:,6)=finalomega(:,6)*180/pi;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
subplot(2,2,2),plot(finaltime,finalomega(:,4),'b-',finaltime,finalomega(:,5),'g--',
                    finaltime,finalomega(:,6),'r:')
title('Simsat rollrate(solid), yawrate(dashed),
pitchrate(dotted)') xlabel('Time (sec)') ylabel('Simsat Angular
Velocity (deg/sec)')
%text(6,-.2,'Yaw rate')
%text(4,.015,'Roll rate')
%text(5,-.01,'Pitch rate')

%Plot euler angles
%Convert to degrees%%%%%%%%%%%%%%%%%%%%
finalomega(:,7)=finalomega(:,7)*180/pi;
finalomega(:,8)=finalomega(:,8)*180/pi;
finalomega(:,9)=finalomega(:,9)*180/pi;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
subplot(2,2,1),plot(finaltime,finalomega(:,7),'b-',finaltime,finalomega(:,8),'g--',
                    finaltime,finalomega(:,9),'r:')
title('Roll(solid), Yaw(dashed), and Pitch(dotted)') xlabel('Time
(sec)') ylabel('Simsat Euler Angles (deg)')


gtext({'SIMSAT Final Baseline Design ','Open-loop "Bang-Bang"
Control '})

%text(6.9,-.9,'Yaw Angle')
%text(4,.055,'Roll Angle')
%text(5,-.05,'Pitch Angle')
```

A.6 domega.m

```
%%%%%%%%%%%%%%
% "domega.m" %
%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 36V Case %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%function domega = domega(t,omega,flag,I1,I2,I3,V,C1,rotaw1,rotaw2,rotaw3,Icomp,Jw1w1d,Jw2w2f,...
%                         Jw3w3h,Jw1cb,Jw2cb,Jw3cb,INVTempmatrix)
function domega =
domega(t,omega,flag,Ctorqw1max,Ctorqw2max,Ctorqw3max,torqdir,rotaw1,rotaw2,rotaw3,Icomp,Jw1w1d,...
Jw2w2f,Jw3w3h,Jw1cb,Jw2cb,Jw3cb,INVTempmatrix)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
% Assumptions                                               %
% 1) No losses in the motor                                 %
% 2) No disturbance torques                                 %
%                                                           %
% Note!:  Rotor inertia of smartmotor has NOT been included %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                                  %%
% Definitions of Terms                                                              %
%                                                                                   %
% Angular Velocity (in rad/sec)                                                     %
%    ww1d - Angular Velocity of wheel one with respect to the d frame               %
%    ww2f - Angular Velocity of wheel two with respect to the f frame               %
%    ww3h - Angular Velocity of wheel three with respect to the h frame             %
%                                                                                   %
% Angular Acceleration (in rad/sec)                                                 %
% ww1ddot - Angular acceleration of wheel one with respect to the d basis           %
% ww2fdot - Angular acceleration of wheel two with respect to the f basis           %
% ww3hdot - Angular acceleration of wheel three with respect to the h basis         %
%                                                                                   %
% Inertia Matrices (in kg*m^2)                                                      %
%  Jw1w1d - Inertia of wheel one about the wheel one center of mass wrt d basis     %
%  Jw2w2f - Inertia of wheel two about the wheel two center of mass wrt f basis     %
%  Jw3w3h - Inertia of wheel three about the wheel three center of mass wrt h basis %
%                                                                                   %
%    Iscb - Simsat inertia matrix about the Simsat center of mass wrt b basis       %
%   Jw1cb - Wheel one inertia matrix about the Simsat center of mass wrt b basis    %
%   Jw2cb - Wheel two inertia matrix about the Simsat center of mass wrt b basis    %
%   Jw3cb - Wheel three inertia matrix about the Simsat center of mass wrt b basis  %
%                                                                                   %
% Transformation Matrices                                                           %
%  rotaw1 - Transformation Matrix for moving from the d basis to the b basis        %
%  rotaw2 - Transformation Matrix for moving from the f basis to the b basis        %
%  rotaw3 - Transformation Matrix for moving from the h basis to the b basis        %
%                                                                                   %
% Miscellaneous Parameters                                                          %
% I1 - I3 - Current for each motor (in Amps)                                        %
%       V - Voltage                (in Volts)                                       %
%                                                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

domega=zeros(9,1);

ww1d(1,1)=0; ww1d(3,1)=0; ww2f(1,1)=0; ww2f(3,1)=0; ww3h(1,1)=0;
ww3h(3,1)=0;

ww1ddot(1,1)=0; ww1ddot(3,1)=0; ww2fdot(1,1)=0; ww2fdot(3,1)=0;
ww3hdot(1,1)=0; ww3hdot(3,1)=0;

ww1d(2,1)=omega(1); ww2f(2,1)=omega(2); ww3h(2,1)=omega(3);

ww1ddot(2,1)=domega(1); ww2fdot(2,1)=domega(2);
ww3hdot(2,1)=domega(3);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                          %
% From Power In = Power Out, we were able to get the following expression  %
% for angular acceleration as a function of current, voltage, wheel torque %
% and angular velocity of the wheel.  C1 is a correction term to prevent   %
% divide by zero errors and keep us in the realm of peak torque.           %
%                                                                          %
% Power In  = (I * V)/w                                                    %
% Power Out = J * wdot                                                     %
%                                                                          %
%  J * wdot = (I * V)/w                                                    %
%      wdot = (I * V)/(J * w)                                              %
%                                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%ww1ddot(2,1)=I1*V/(Jw1w1d(2,2)*(ww1d(2,1)+C1));
%ww2fdot(2,1)=I2*V/(Jw2w2f(2,2)*(ww2f(2,1)+C1));
%ww3hdot(2,1)=I3*V/(Jw3w3h(2,2)*(ww3h(2,1)+C1));
```

```
%J*alpha=Torque, alpha=Torque/J, using 36V Torque curve
if torqdir==1
   Torquew1=0;
   %Torquew1=Ctorqw1max-exp(.0193942*ww1d(2,1));
   %Torquew1=.007061*Torquew1;
   %Torquew2=0;
   Torquew2=Ctorqw2max-exp(.0193942*ww2f(2,1));
   Torquew2=.007061*Torquew2; %convert from oz-in to N-m
    Torquew3=0;
   %Torquew3=Ctorqw3max-exp(.0193942*ww3h(2,1));
   %Torquew3=.007061*Torquew3;
end

if torqdir==2
   Torquew1=0;
   %Torquew1=-Ctorqw1max+exp(.0193942*ww1d(2,1));
   %Torquew1=.007061*Torquew1;
   %Torquew2=0;
   Torquew2=-Ctorqw2max+exp(.0193942*ww2f(2,1));
   Torquew2=.007061*Torquew2; %convert from oz-in to N-m
    Torquew3=0;
   %Torquew3=-Ctorqw3max+exp(.0193942*ww3h(2,1));
   %Torquew3=.007061*Torquew3;
end

ww1ddot(2,1)=Torquew1/Jw1w1d(2,2);
ww2fdot(2,1)=Torquew2/Jw2w2f(2,2);
ww3hdot(2,1)=Torquew3/Jw3w3h(2,2);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
% For this section, we are solving for the second part of our system           %
% term by term.  To understand what we are talking about, review the           %
% equations of motion within the ADACS portion of the detailed design chapter. %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

term1=rotaw1*(Jw1w1d*ww1ddot(1:3));
```

```
term2=rotaw2*(Jw2w2f*ww2fdot(1:3));

term3=rotaw3*(Jw3w3h*ww3hdot(1:3));

temp4=Icomp*omega(4:6); term4=cross(omega(4:6),temp4);

temp5=Jw1cb*omega(4:6); term5=cross(omega(4:6),temp5);

temp6=rotaw1*(Jw1w1d*ww1d); term6=cross(omega(4:6),temp6);

temp7=Jw2cb*omega(4:6); term7=cross(omega(4:6),temp7);

temp8=rotaw2*(Jw2w2f*ww2f); term8=cross(omega(4:6),temp8);

temp9=Jw3cb*omega(4:6); term9=cross(omega(4:6),temp9);

temp10=rotaw3*(Jw3w3h*ww3h); term10=cross(omega(4:6),temp10);


Secondportion=term1+term2+term3+term4+term5+term6+term7+term8+term9+term10;

%Calculate equations of motion by finding omegadot**************************************************

domega(1)=ww1ddot(2,1); domega(2)=ww2fdot(2,1);
domega(3)=ww3hdot(2,1);

domega(4:6)=INVTempmatrix*Secondportion;

%Euler angles for roll angle, yaw angle, and pitch angle%
%domega(7) =omega(4)*(cos(omega(9))/cos(omega(8))) + omega(6)*(sin(omega(9))/cos(omega(8)));
%domega(8) =-sin(omega(9))*omega(4) + cos(omega(9))*omega(6);
%domega(9) =tan(omega(8))*cos(omega(9))*omega(4) + omega(5) + tan(omega(8))*sin(omega(9))*omega(6);

domega(7) = omega(4) + omega(5)*-cos(omega(7))*tan(omega(9)) +
omega(6)*sin(omega(7))*tan(omega(9)); domega(8) =
omega(5)*(cos(omega(7))/cos(omega(9))) -
omega(6)*(sin(omega(7))/cos(omega(9))); domega(9) =
omega(5)*sin(omega(7)) + omega(6)*cos(omega(7));
```

Appendix B. Dynamic Motor Testing

The researcher conducted dynamic testing on each motor to determine response times to step inputs. The desired response time was one second but the average response time achieved for each motor was 1.5 seconds. When the motor begins from rest, or when the input voltage changes sign, a slightly longer response time is evident. The following figures illustrate the various response times for each motor when experiencing a voltage step input.

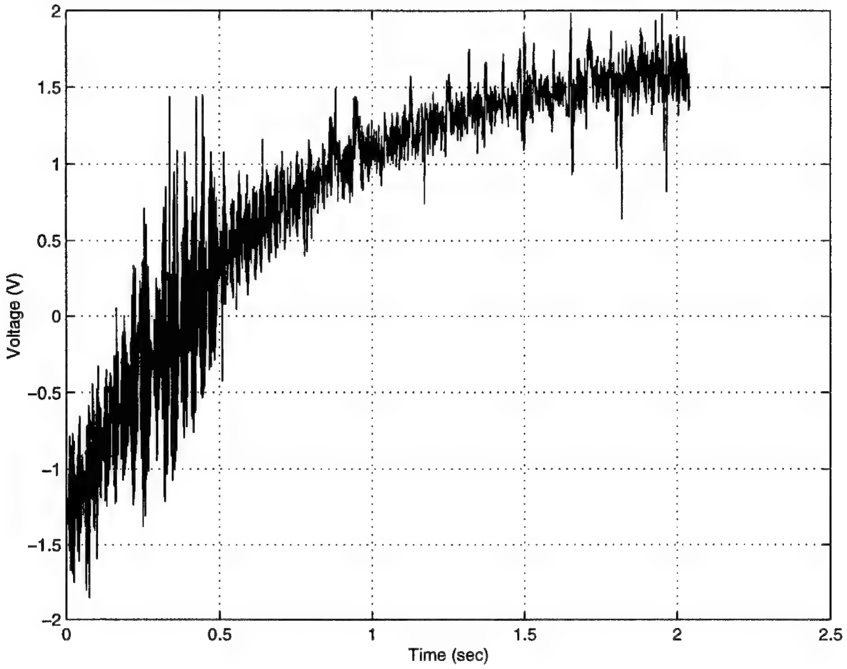


Figure B.1 Motor #1 Step Response -1 Volt to 1 Volt

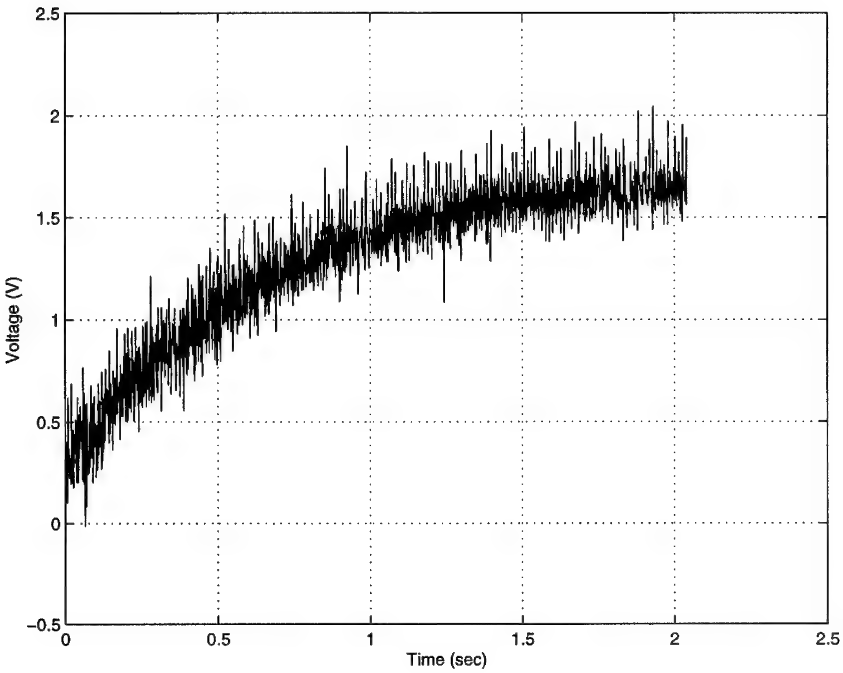


Figure B.2 Motor #1 Step Response 0 Volts to 1 Volt

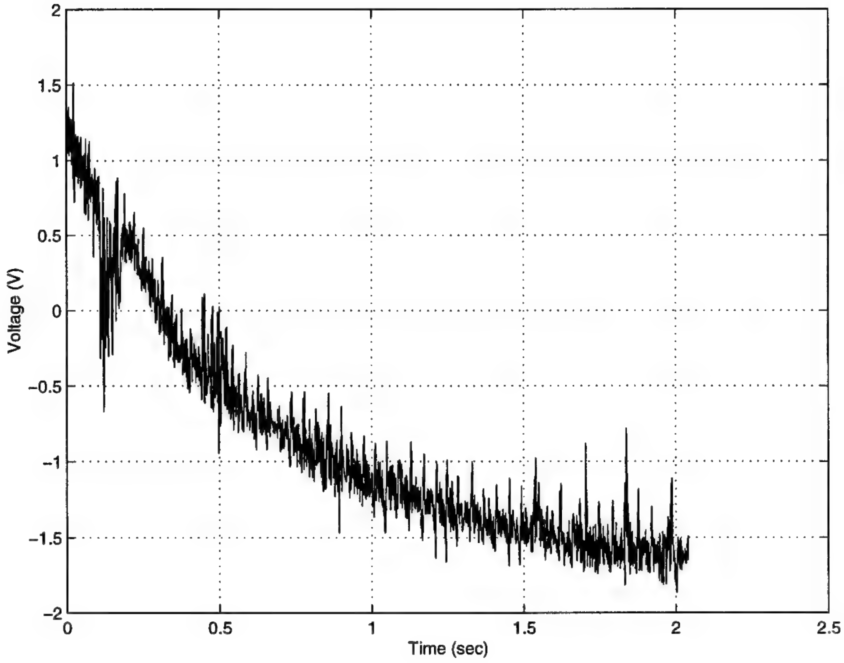


Figure B.3 Motor #1 Step Response 1 Volt to -1 Volt

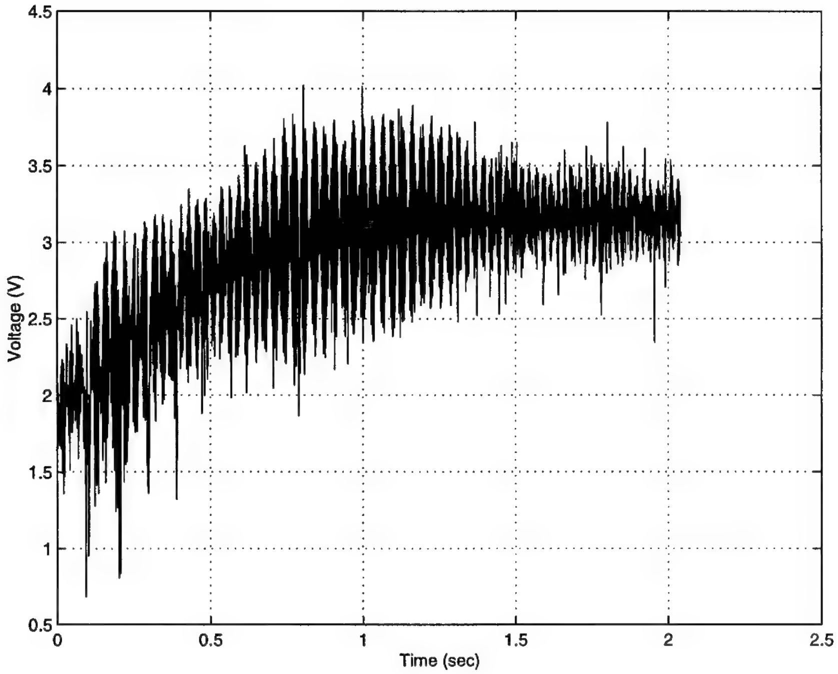


Figure B.4 Motor #1 Step Response 1 Volt to 2 Volts

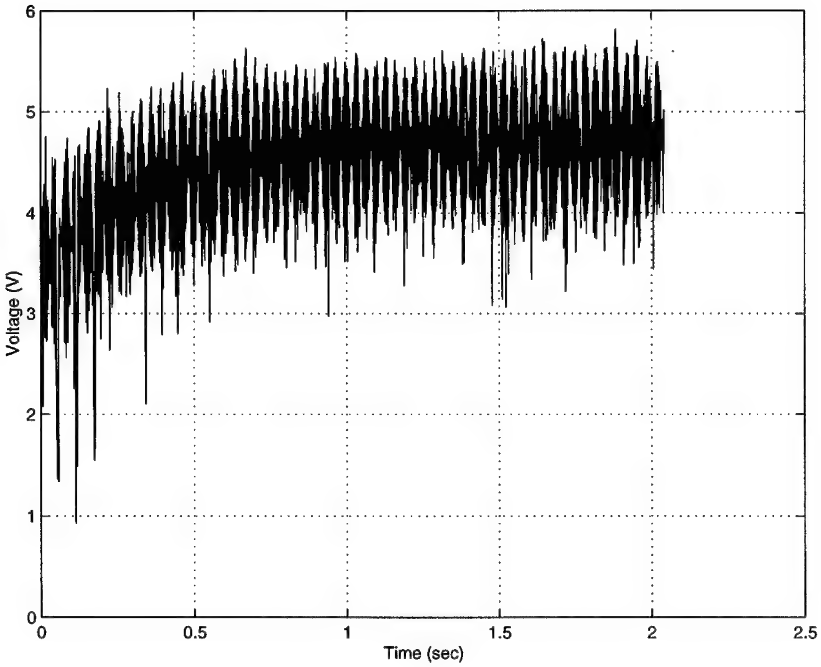


Figure B.5 Motor #1 Step Response 2 Volts to 3 Volts

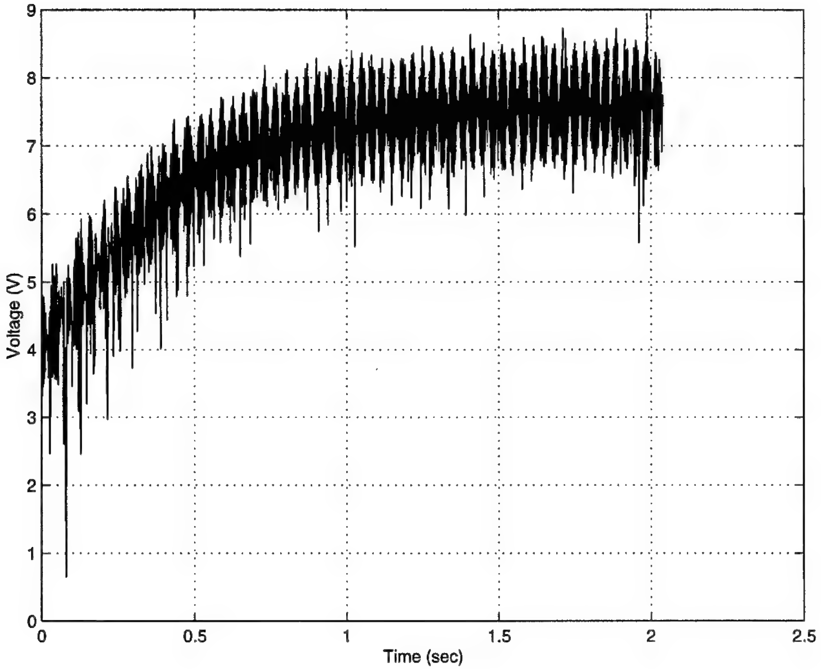


Figure B.6 Motor #1 Step Response 2 Volts to 5 Volts

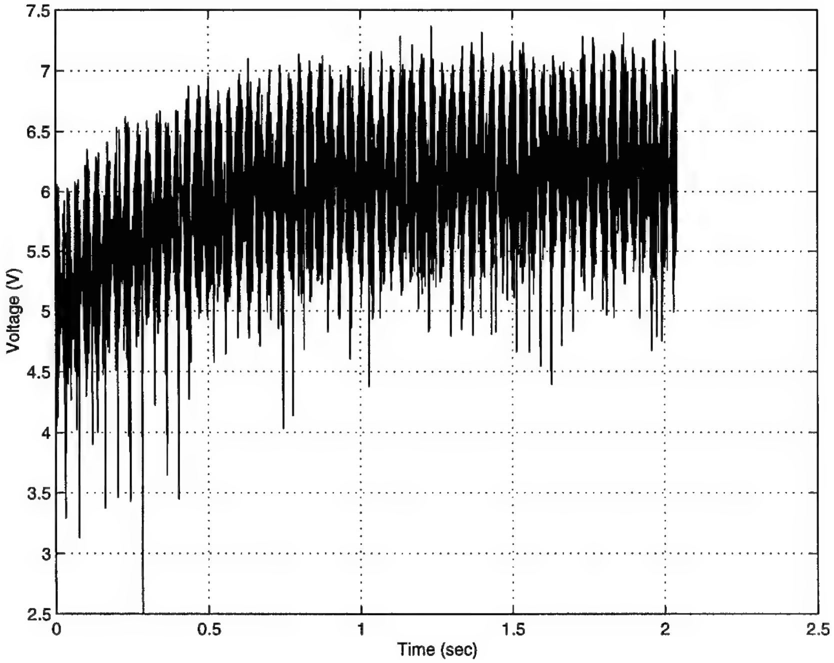


Figure B.7 Motor #1 Step Response 3 Volts to 4 Volts

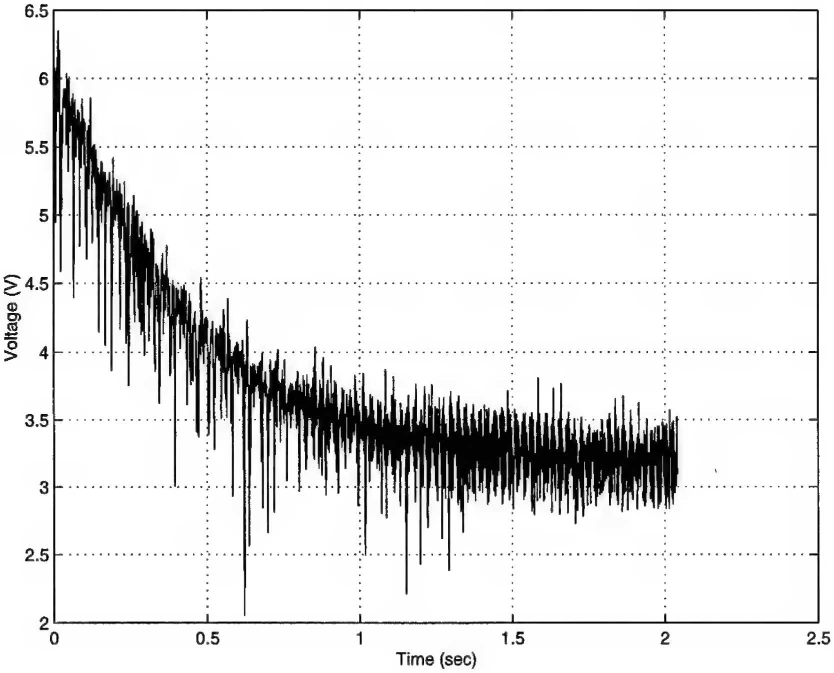


Figure B.8 Motor #1 Step Response 4 Volts to 2 Volts

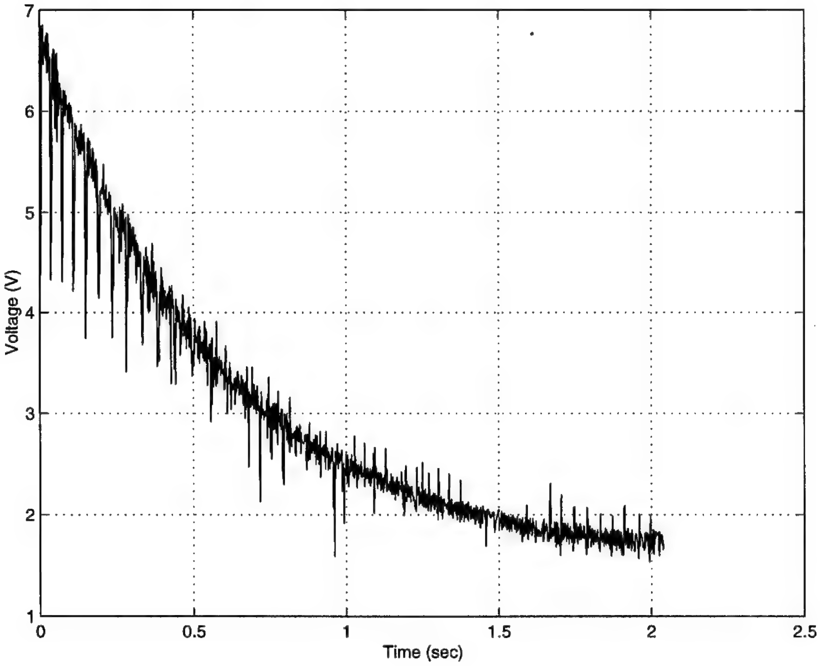


Figure B.9 Motor #1 Step Response 5 Volts to 1 Volt

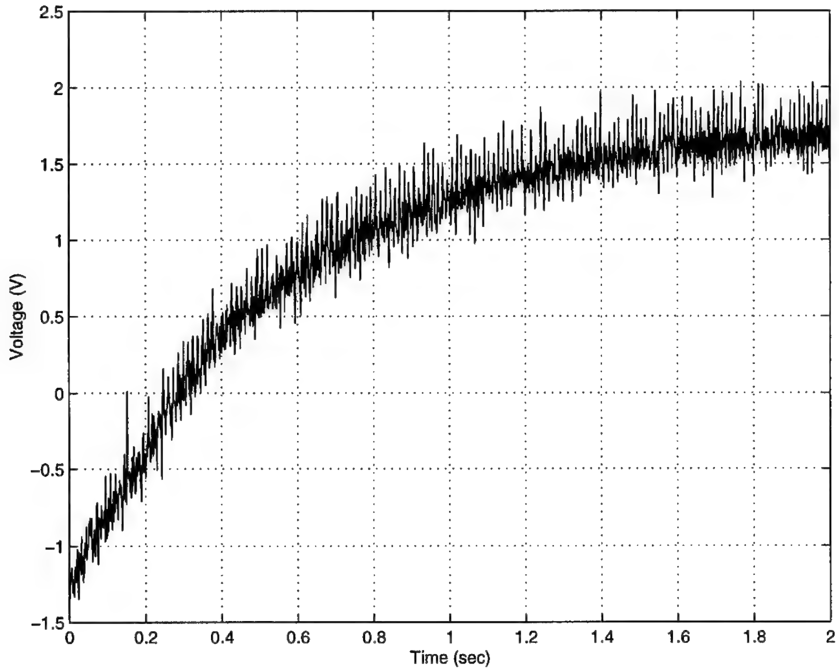


Figure B.10 Motor #2 Step Response -1 Volt to 1 Volt

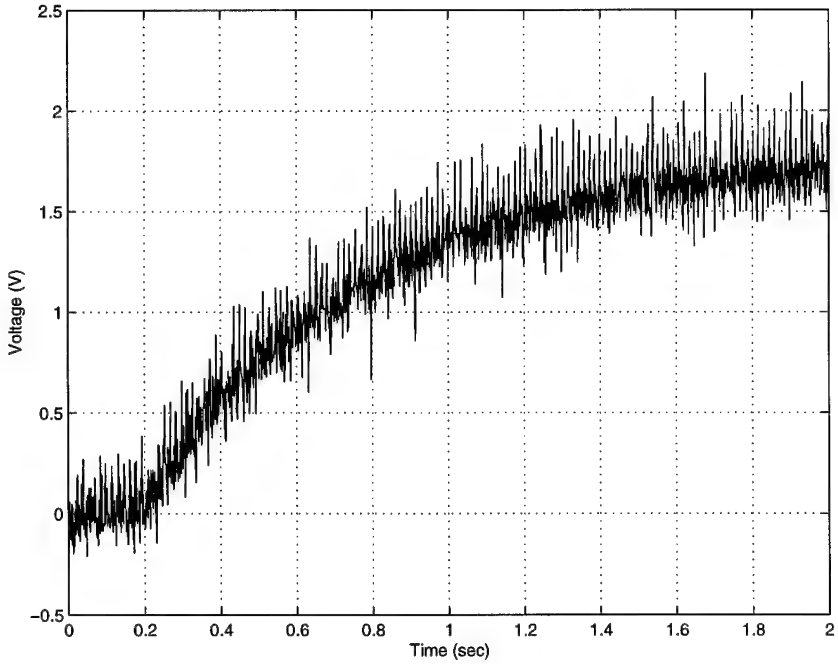


Figure B.11 Motor #2 Step Response 0 Volts to 1 Volt

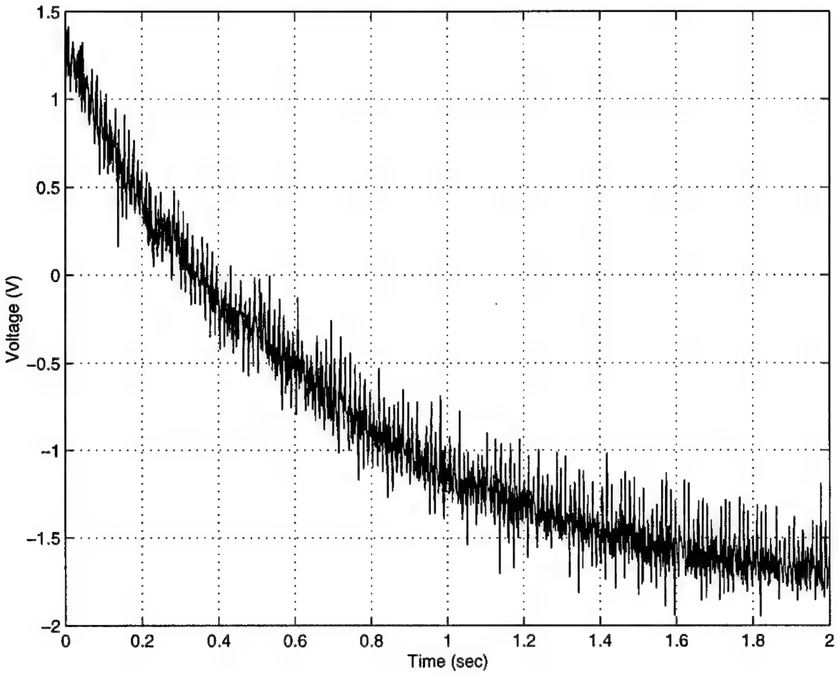


Figure B.12 Motor #2 Step Response 1 Volt to -1 Volt

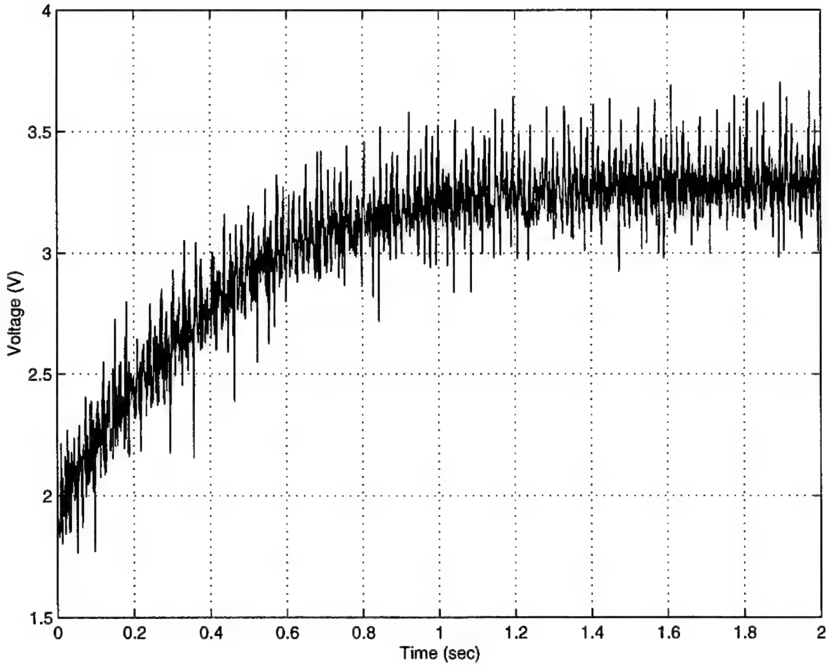


Figure B.13 Motor #2 Step Response 1 Volt to 2 Volts

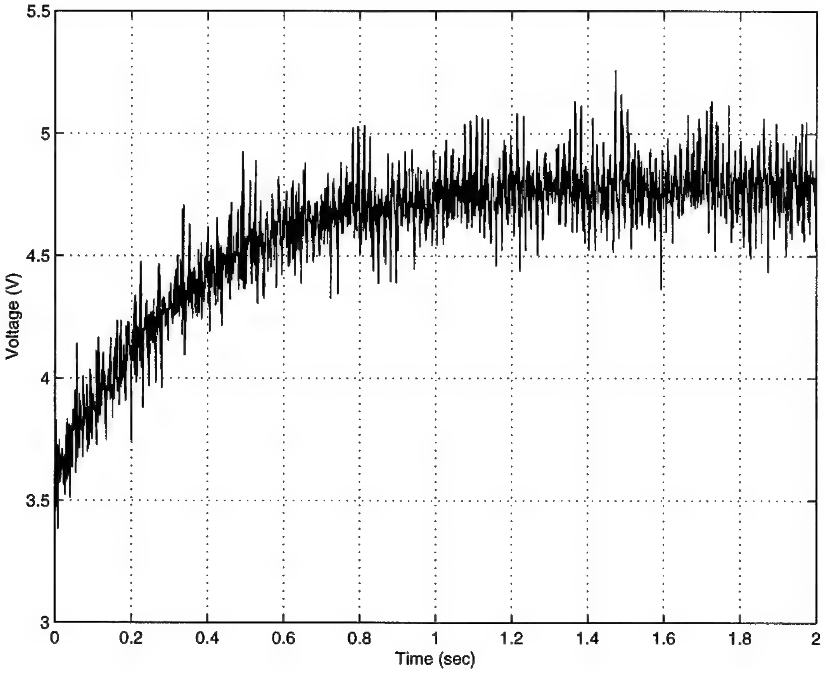


Figure B.14 Motor #2 Step Response 2 Volts to 3 Volts

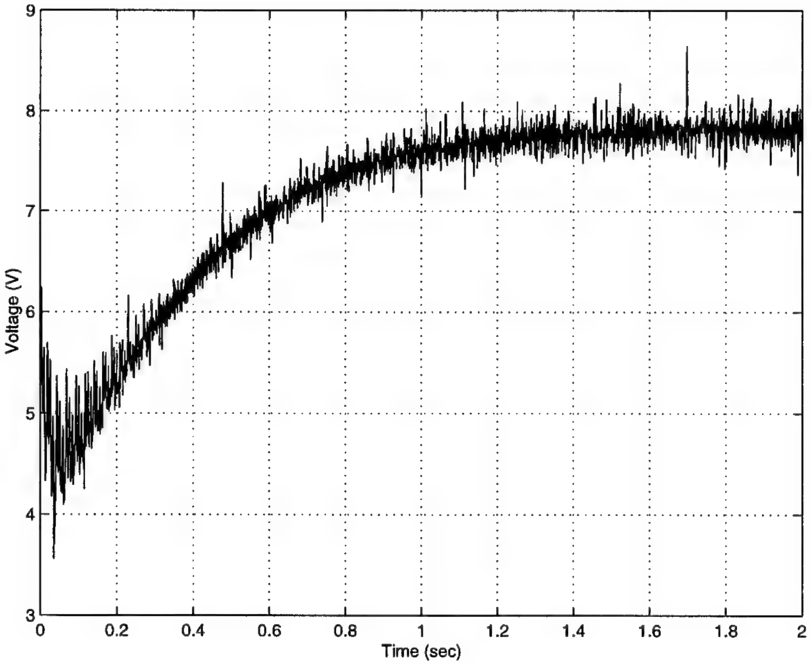


Figure B.15 Motor #2 Step Response 2 Volts to 5 Volts

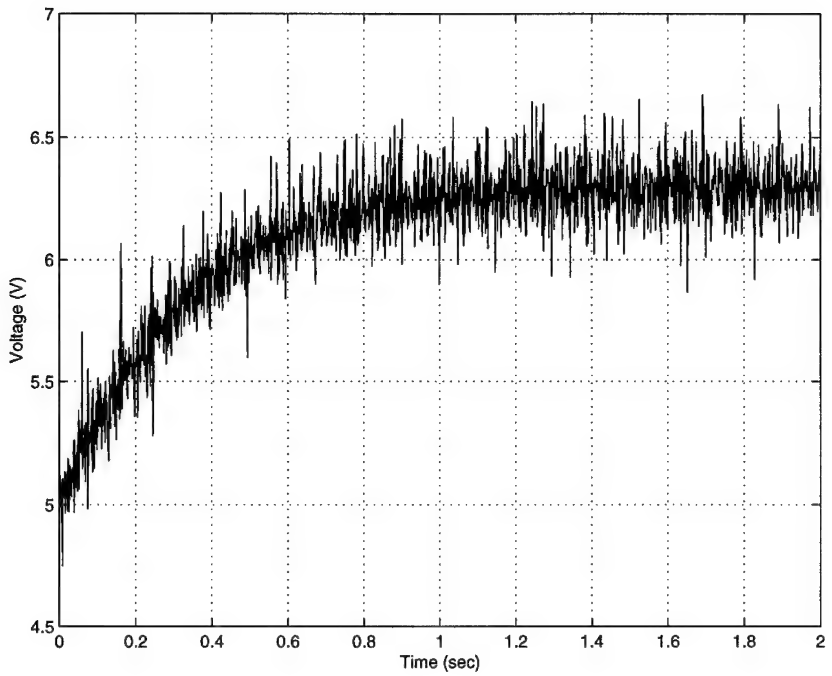


Figure B.16 Motor #2 Step Response 3 Volts to 4 Volts

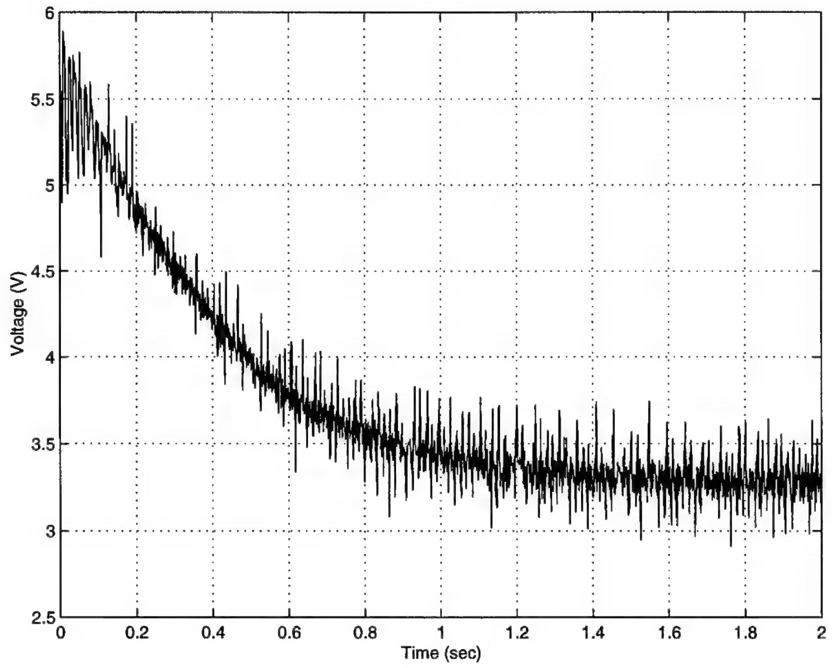


Figure B.17 Motor #2 Step Response 4 Volts to 2 Volts

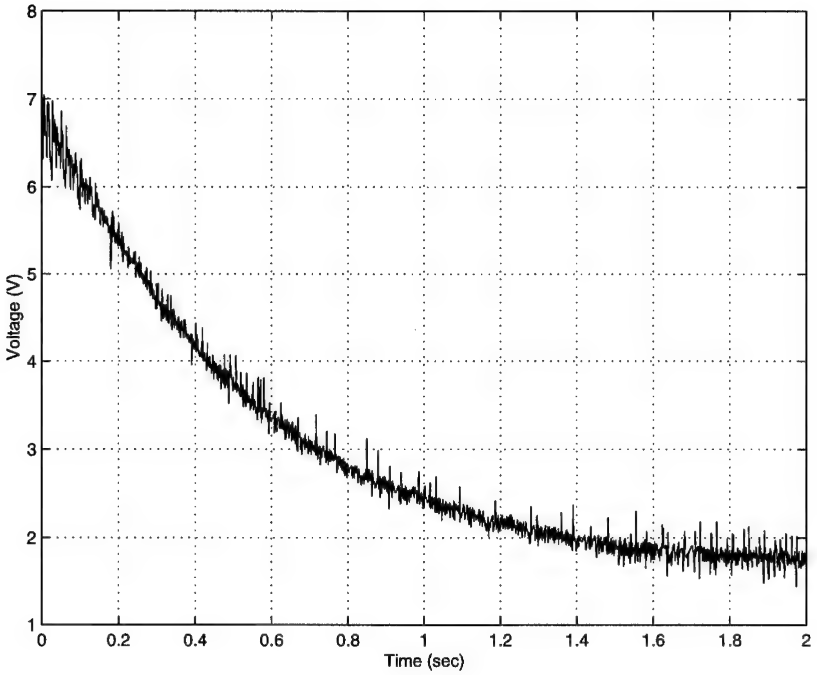


Figure B.18 Motor #2 Step Response 5 Volts to 1 Volt

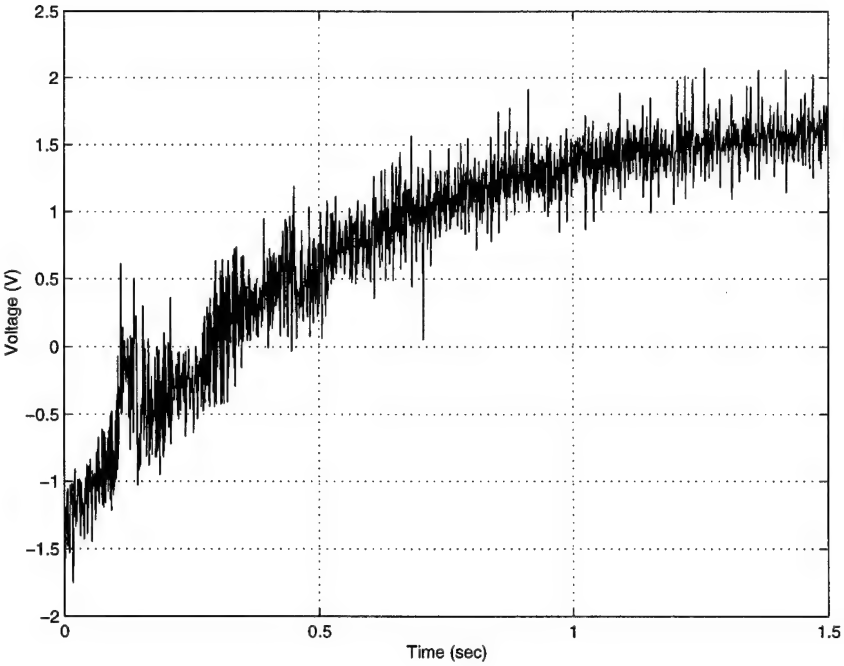


Figure B.19 Motor #3 Step Response -1 Volt to 1 Volt

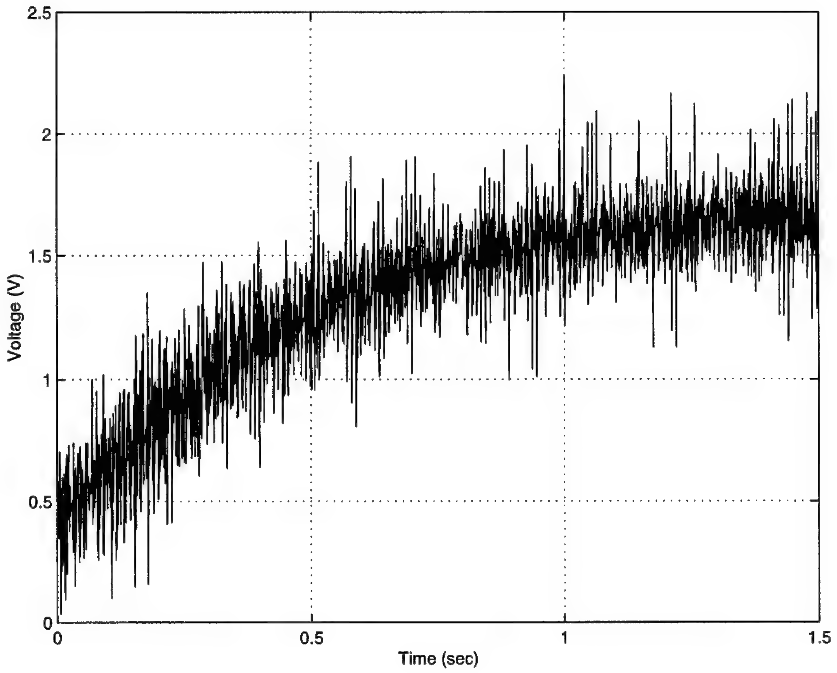


Figure B.20 Motor #3 Step Response 0 Volts to 1 Volt

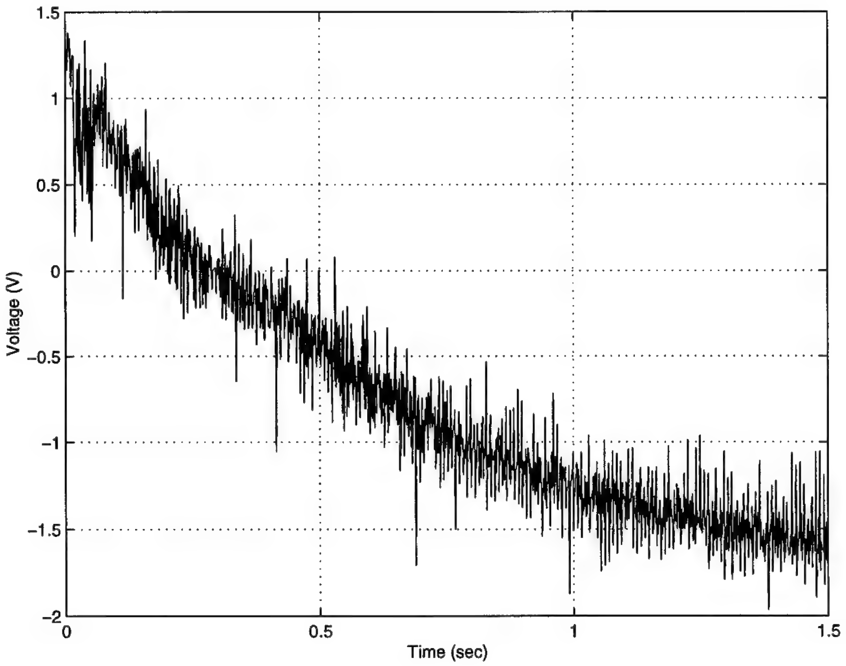


Figure B.21 Motor #3 Step Response 1 Volt to -1 Volt

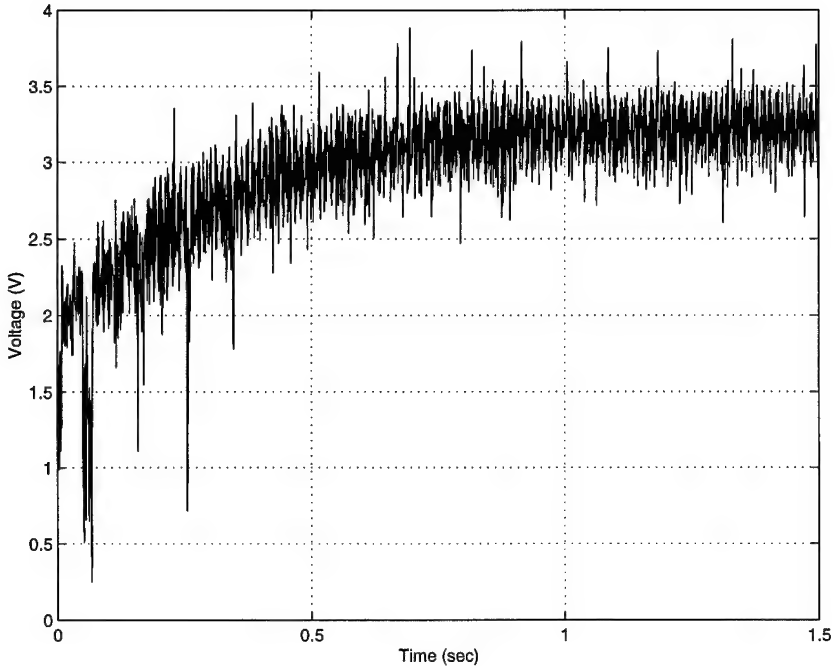


Figure B.22 Motor #3 Step Response 1 Volt to 2 Volts

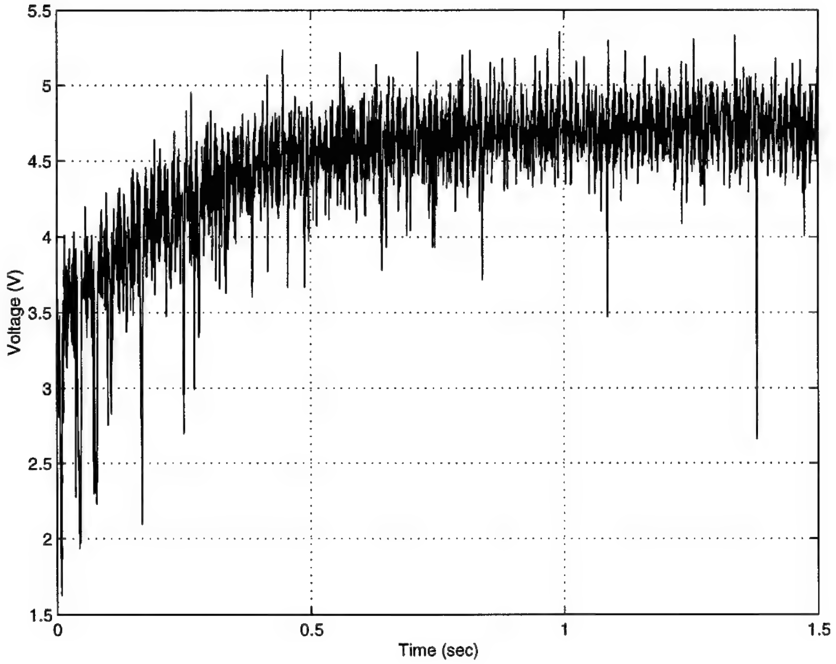


Figure B.23 Motor #3 Step Response 2 Volts to 3 Volts

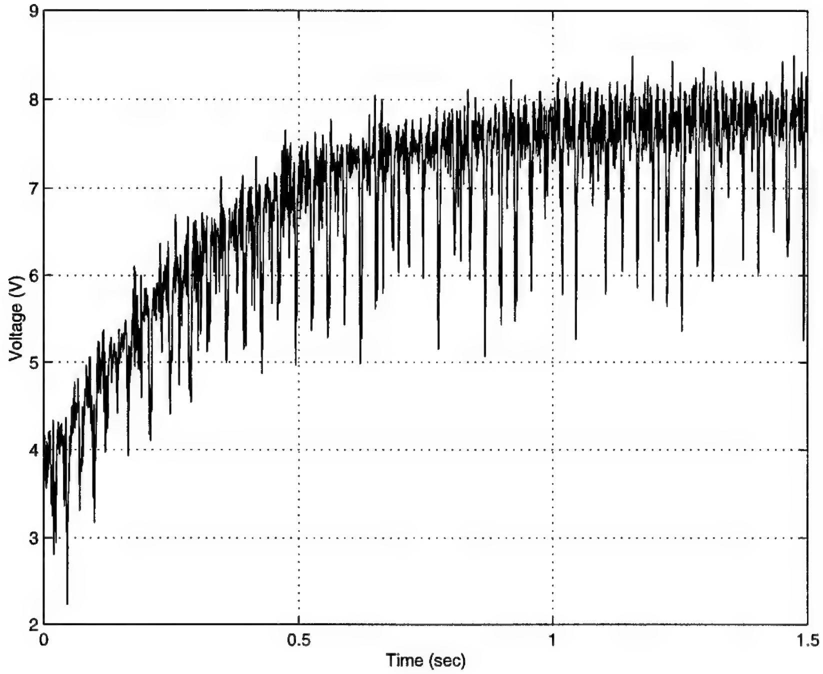


Figure B.24 Motor #3 Step Response 2 Volts to 5 Volts

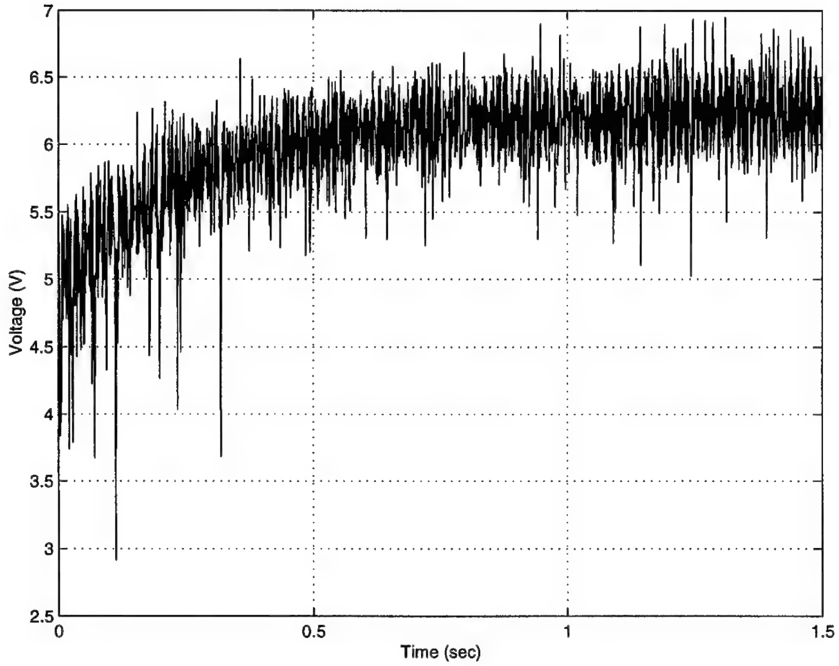


Figure B.25 Motor #3 Step Response 3 Volts to 4 Volts

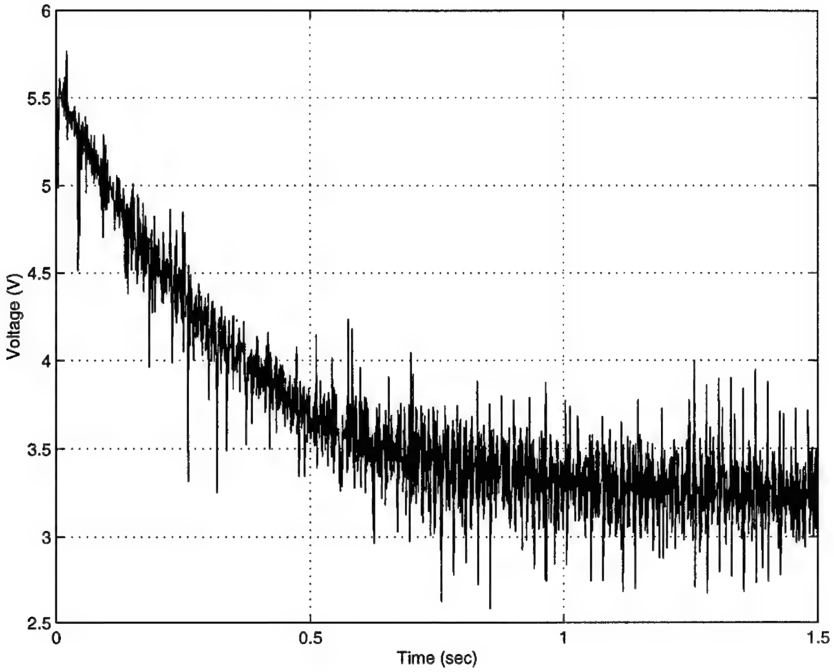


Figure B.26 Motor #3 Step Response 4 Volts to 2 Volts

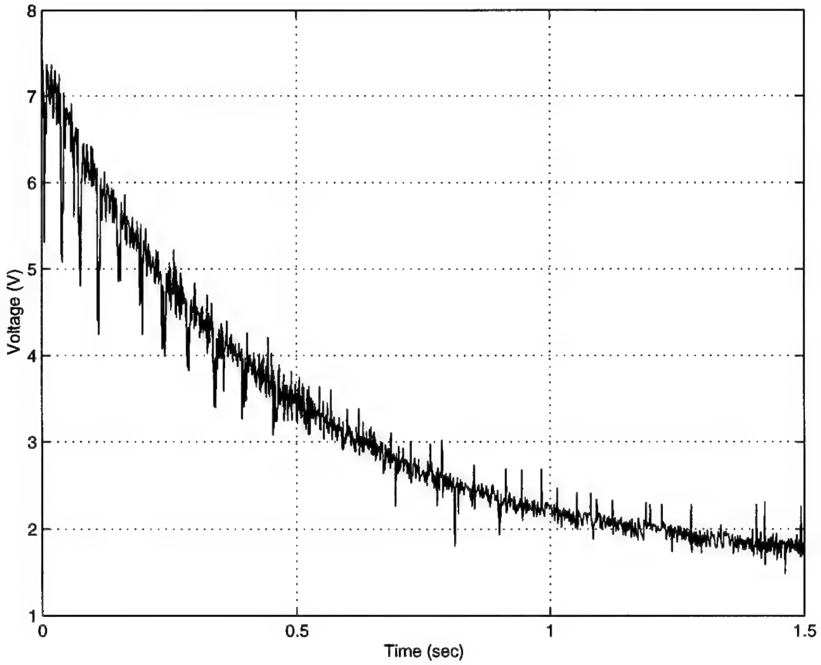


Figure B.27 Motor #3 Step Response 5 Volts to 1 Volt

Appendix C. Gyro Calibration Curves

This appendix contains all of the angular velocity and acceleration information provided by the manufacturer of the Humphrey CF75 Gyro. The following calculations and plots were generated by AFIT's 1999 Graduate Systems Engineering Team [7].

C.1 Pitch Axis Regression

Best Fit Curve Formula: $PitchRate = 16.01418821 * OutputVoltage - 40.13635991$

Table C.1 Pitch Axis Gyro Regression Analysis

Output Voltage	Truth	Pitch Rate	Delta
0.006	-40	-40.04027478	0.040274781
0.008	-40	-40.0082464	0.008246404
0.508	-32	-32.0011523	0.001152299
0.510	-32	-31.96912392	-0.030876077
1.010	-24	-23.96202982	-0.037970182
1.012	-24	-23.93000144	-0.069998559
1.504	-16	-16.05102084	0.051020842
1.511	-16	-15.93892152	-0.061078475
2.006	-8	-8.011898361	0.011898361
2.009	-8	-7.963855796	-0.036144204
2.505	0	-0.020818444	0.020818444
2.507	0	0.011209932	-0.011209932
3.001	8	7.922218908	0.077781092
3.005	8	7.986275661	0.013724339
3.506	16	16.00938395	-0.009383954
3.509	16	16.05742652	-0.057426519
4.009	24	24.06452062	-0.064520624
4.012	24	24.11256319	-0.112563189
4.506	32	32.02357216	-0.023572164
4.509	32	32.07161473	-0.071614729
5.014	40	40.15877977	-0.158779775
5.016	40	40.19080815	-0.190808151

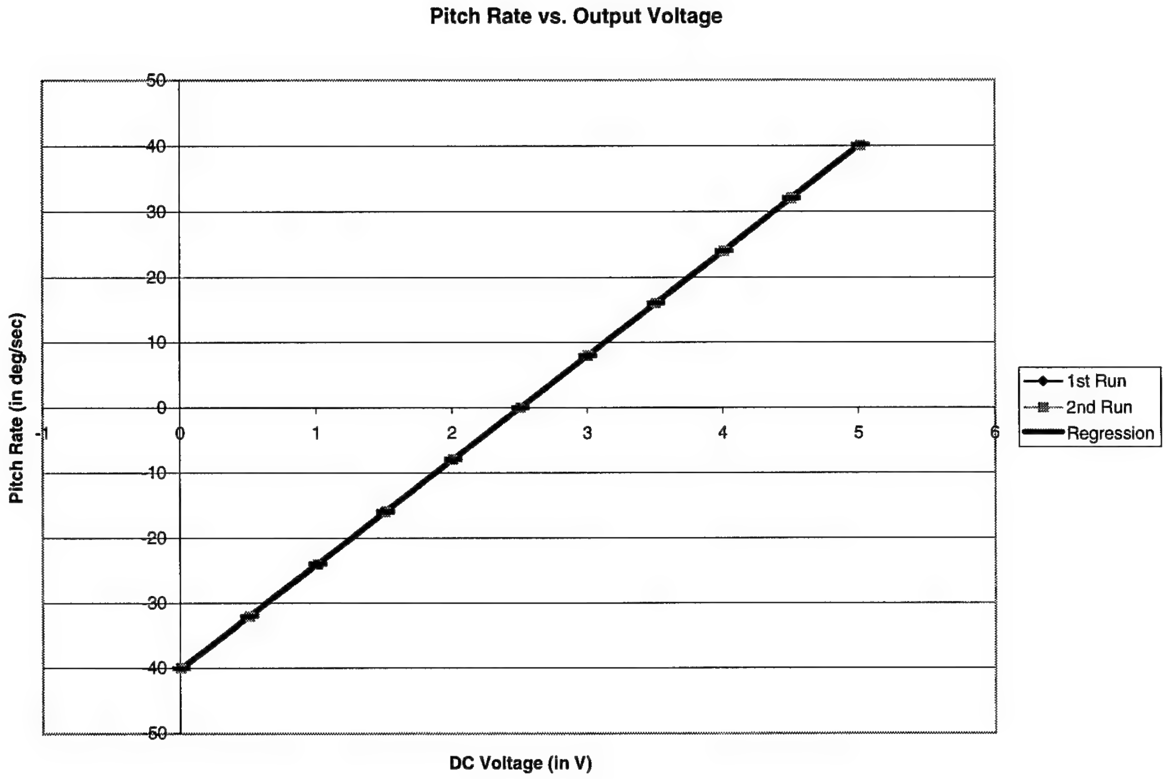


Figure C.1 Pitch Rate vs. Output Voltage

C.2 Roll Axis Regression

Best Fit Curve Formula: $RollRate = 47.67069299 * OutputVoltage - 119.2577727$

Table C.2 Roll Axis Gyro Regression Analysis

Output Voltage	Truth	Roll Rate	Delta
-0.015	-120	-119.9728331	-0.027166905
-0.012	-120	-119.829821	-0.170178984
0.490	-96	-95.89913313	-0.100866865
0.492	-96	-95.80379175	-0.196208251
0.990	-72	-72.06378664	0.06378664
0.992	-72	-71.96844525	-0.031554746
1.495	-48	-47.99008668	-0.00991332
1.498	-48	-47.8470746	-0.152925399
1.996	-24	-24.10706949	0.107069492
1.998	-24	-24.01172811	0.011728106
2.500	0	-0.081040225	0.081040225
2.502	0	0.014301161	-0.014301161
3.002	24	23.84964766	0.150352344
3.005	24	23.99265973	0.007340265
3.509	48	48.018689	-0.018689002
3.512	48	48.16170108	-0.161701081
4.011	72	71.94937688	0.050623117
4.015	72	72.14005965	-0.140059655
4.511	96	95.78472338	0.215276622
4.513	96	95.88006476	0.119935236
5.010	120	119.5723992	0.42760082
5.012	120	119.6677406	0.332259434

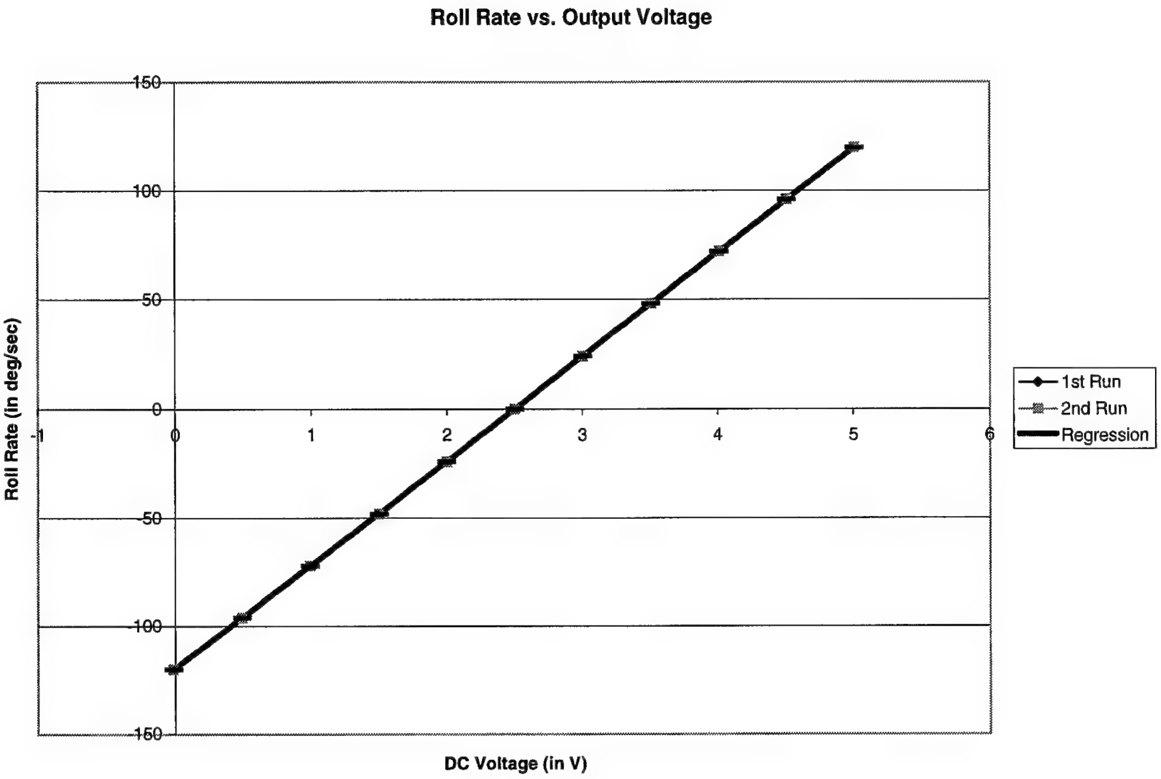


Figure C.2 Roll Rate vs. Output Voltage

C.3 Yaw Axis Regression

Best Fit Curve Formula: $YawRate = 16.07699608 * OutputVoltage - 40.28734448$

Table C.3 Yaw Axis Gyro Regression Analysis

Output Voltage	Truth	Yaw Rate	Delta
0.017	-40	-40.01403555	0.014035547
0.019	-40	-39.98188155	-0.018118446
0.509	-32	-32.10415348	0.104153475
0.512	-32	-32.05592249	0.055922487
1.010	-24	-24.04957844	0.049578439
1.012	-24	-24.01742445	0.017424447
1.508	-16	-16.04323439	0.043234391
1.509	-16	-16.0271574	0.027157395
2.008	-8	-8.004736351	0.004736351
2.013	-8	-7.924351371	-0.075648629
2.507	0	0.017684693	-0.017684693
2.509	0	0.049838685	-0.049838685
3.001	8	7.959720756	0.040279244
3.002	8	7.975797752	0.024202248
3.500	16	15.9821418	0.0178582
3.502	16	16.01429579	-0.014295792
4.000	24	24.02063984	-0.02063984
4.003	24	24.06887083	-0.068870828
4.508	32	32.18775385	-0.187753849
4.511	32	32.23598484	-0.235984837
5.012	40	40.29055987	-0.290559873
5.014	40	40.32271387	-0.322713865

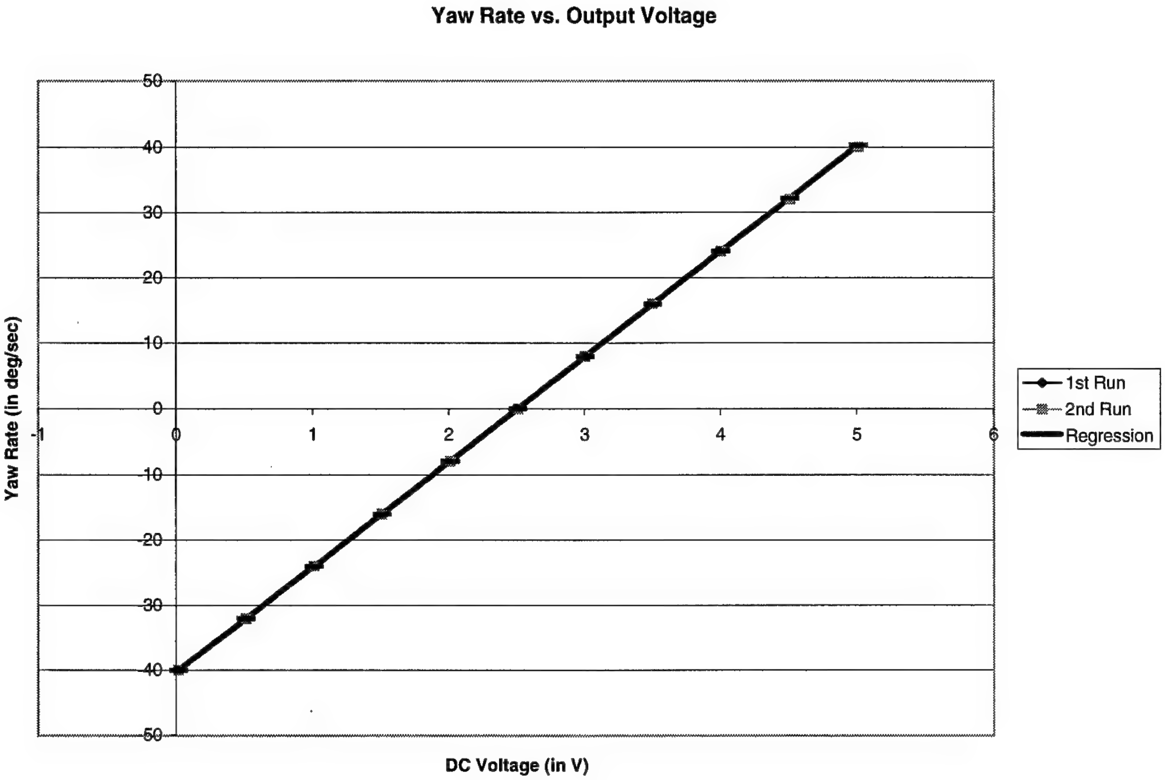


Figure C.3 Yaw Rate vs. Output Voltage

C.4 Fore/Aft Acceleration Regression

Best Fit Curve Formula: $Acceleration = 1.985948707 * OutputVoltage - 4.967038257$

Table C.4 Fore/Aft Acceleration Regression Analysis

Output Voltage	Truth	Measured Gs	Delta
0.005	-5	-4.957108513	-0.042891487
0.005	-5	-4.957108513	-0.042891487
0.455	-4	-4.063431595	0.063431595
0.481	-4	-4.011796929	0.011796929
0.956	-3	-3.068471293	0.068471293
1.012	-3	-2.957258166	-0.042741834
1.457	-2	-2.073510991	0.073510991
1.517	-2	-1.954354068	-0.045645932
1.958	-1	-1.078550689	0.078550689
2.048	-1	-0.899815305	-0.100184695
2.464	0	-0.073660643	0.073660643
2.494	0	-0.014082182	0.014082182
2.985	1	0.961018633	0.038981367
3.055	1	1.100035043	-0.100035043
3.500	2	1.983782218	0.016217783
3.551	2	2.085065602	-0.085065602
3.996	3	2.968812776	0.031187224
4.056	3	3.087969699	-0.087969699
4.512	4	3.993562309	0.006437691
4.542	4	4.05314077	-0.05314077
4.987	5	4.936887945	0.063112055
4.988	5	4.938873894	0.061126106

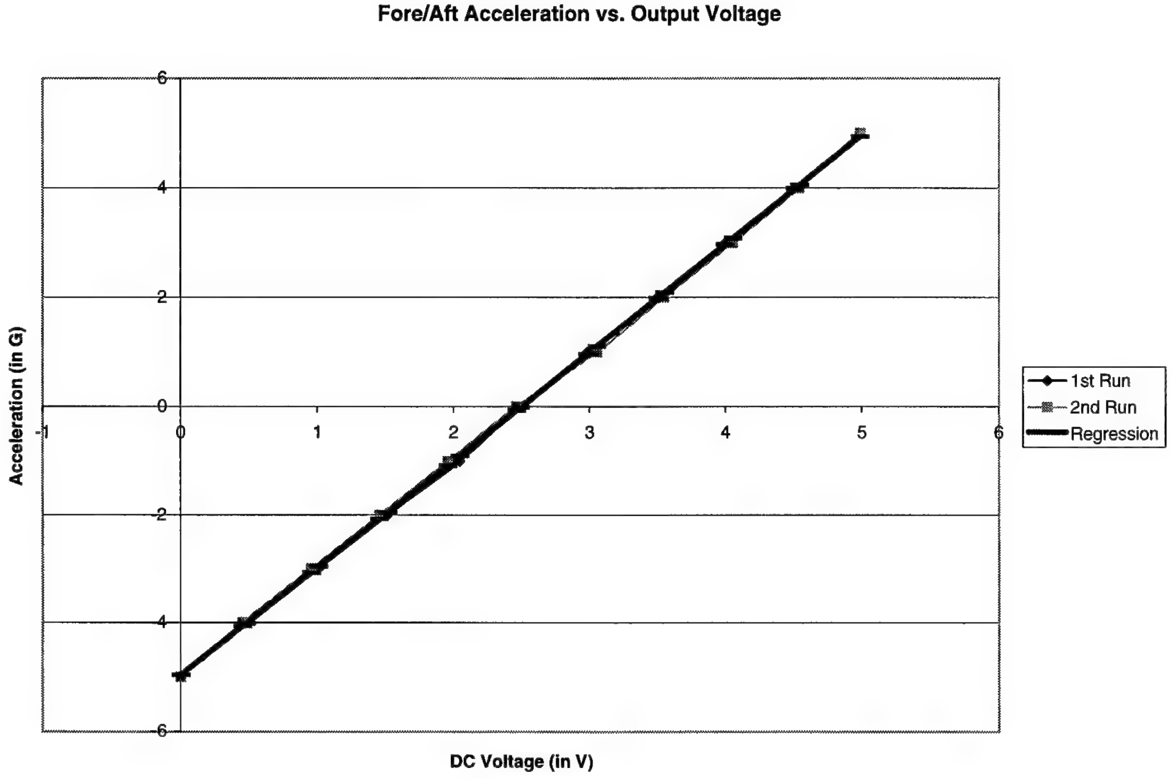


Figure C.4 Fore/Aft Acceleration vs. Output Voltage

C.5 Lateral Acceleration Regression

Best Fit Curve Formula: $Acceleration = 2.004813213 * OutputVoltage - 5.001917838$

Table C.5 Lateral Acceleration Regression Analysis

Output Voltage	Truth	Measured Gs	Delta
0.005	-5	-4.991893772	-0.008106228
0.005	-5	-4.991893772	-0.008106228
0.481	-4	-4.037602683	0.037602683
0.511	-4	-3.977458286	-0.022541714
0.981	-3	-3.035196076	0.035196076
1.012	-3	-2.973046866	-0.026953134
1.457	-2	-2.080904987	0.080904987
1.537	-2	-1.92051993	-0.07948007
1.978	-1	-1.036397303	0.036397303
2.033	-1	-0.926132576	-0.073867424
2.464	0	-0.062058081	0.062058081
2.489	0	-0.011937751	0.011937751
2.975	1	0.962401471	0.037598529
3.035	1	1.082690263	-0.082690263
3.470	2	1.954784011	0.045215989
3.521	2	2.057029485	-0.057029485
3.976	3	2.969219497	0.030780503
4.011	3	3.039387959	-0.039387959
4.477	4	3.973630917	0.026369083
4.507	4	4.033775313	-0.033775313
4.982	5	4.986061589	0.013938411
4.982	5	4.986061589	0.013938411

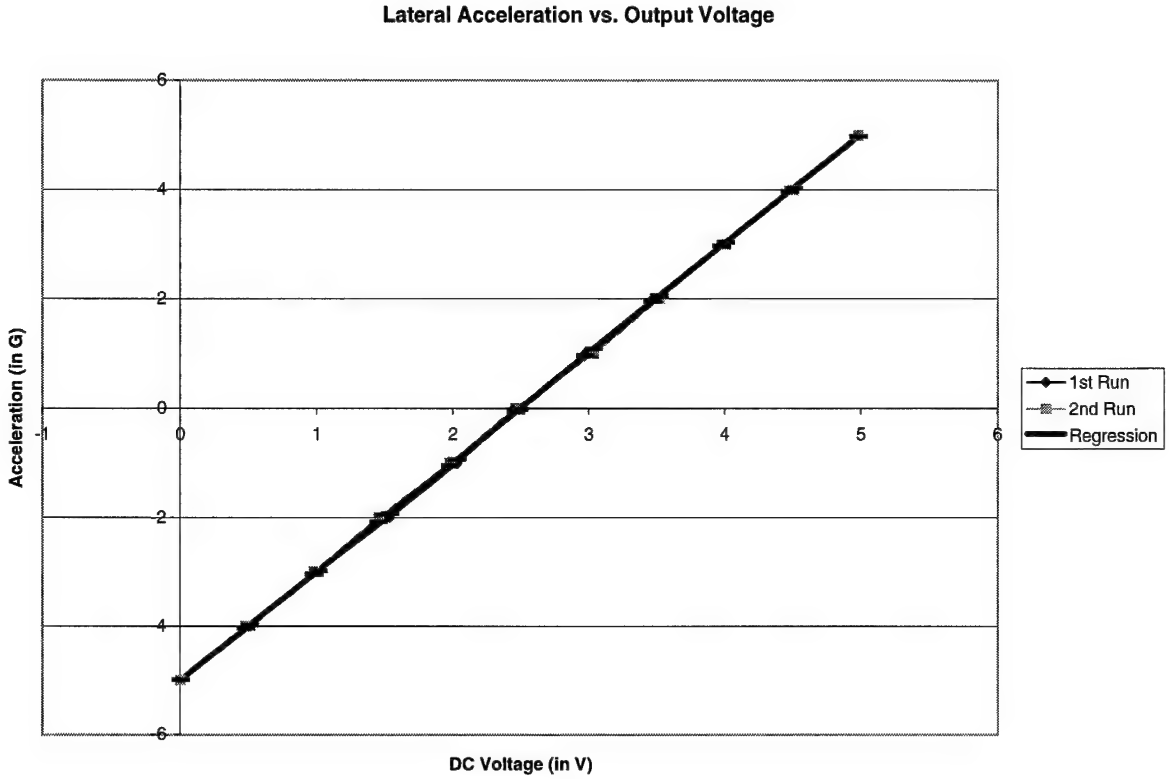


Figure C.5 Lateral Acceleration vs. Output Voltage

C.6 Vertical Acceleration Regression

Best Fit Curve Formula: $Acceleration = 1.990529233 * OutputVoltage - 4.975418296$

Table C.6 Vertical Acceleration Regression Analysis

Output Voltage	Truth	Measured Gs	Delta
0.005	-5	-4.96546565	-0.03453435
0.005	-5	-4.96546565	-0.03453435
0.486	-4	-4.008021089	0.008021089
0.521	-4	-3.938352566	-0.061647434
0.946	-3	-3.092377642	0.092377642
0.991	-3	-3.002803826	0.002803826
1.452	-2	-2.08516985	0.08516985
1.507	-2	-1.975690742	-0.024309258
1.948	-1	-1.09786735	0.09786735
2.043	-1	-0.908767073	-0.091232927
2.459	0	-0.080706912	0.080706912
2.504	0	0.008866903	-0.008866903
2.985	1	0.966311465	0.033688535
3.050	1	1.095695865	-0.095695865
3.490	2	1.971528727	0.028471273
3.546	2	2.082998364	-0.082998364
3.986	3	2.958831227	0.041168773
4.041	3	3.068310335	-0.068310335
4.502	4	3.985944311	0.014055689
4.527	4	4.035707542	-0.035707542
4.998	5	4.973246811	0.026753189
4.998	5	4.973246811	0.026753189

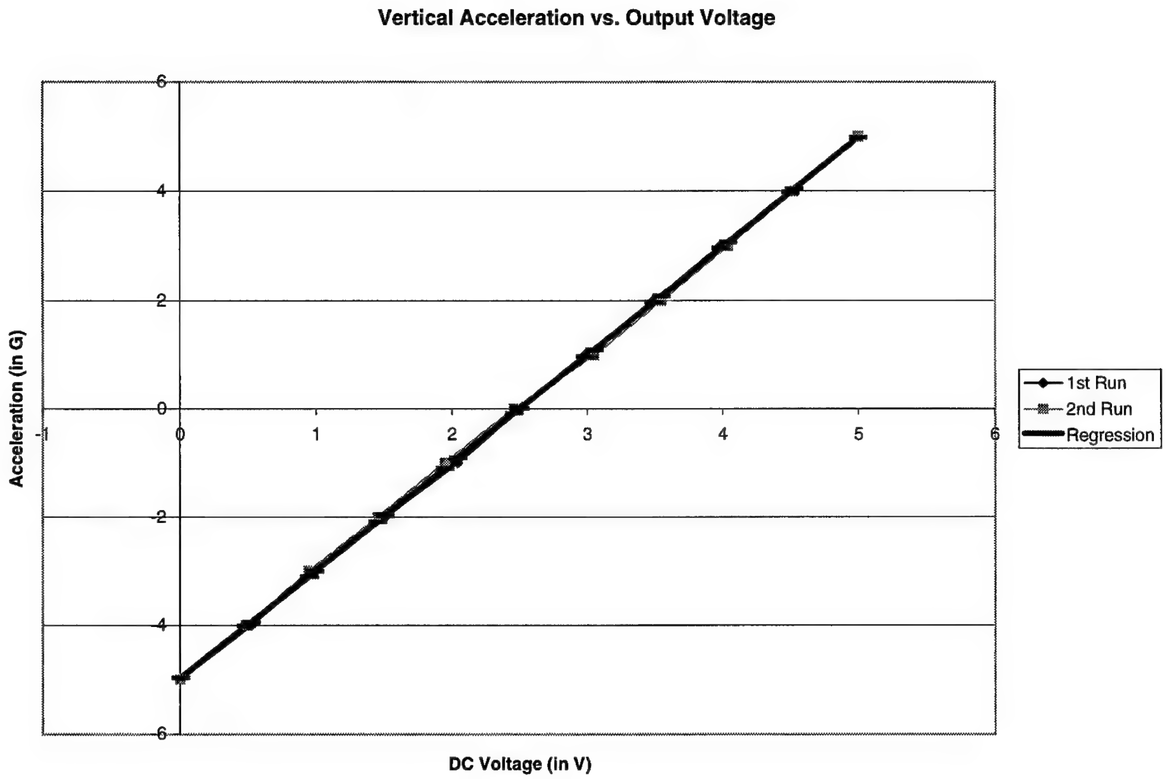


Figure C.6 Vertical Acceleration vs. Output Voltage

Appendix D. Multimedia Lesson Plan

D.1 Lesson Plan

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright Patterson Air Force Base, Ohio

COVER SHEET

LESSON TITLE: Conservation of Angular Momentum

METHOD: Formal Lecture

SLIDES: lesson1.ppt

PART I

COGNITIVE OBJECTIVE: Know how Newton's Second Law relates to angular momentum, how Newton's Third Law describes conservation of angular momentum, and the roational coupling effects of an isolated rigid body.

COGNITIVE SAMPLES OF BEHAVIOR:

1a. Identify Newton's Second Law.

b. Derive the equations for angular momentum and torque.

2a. Identify Newton's Third Law.

b. State why angular momentum is conserved in an isolated system.

3a. Identify Euler's rotational equations of motion.

b. State why rotational coupling occurs.

c. Describe the two coupling cases for an axisymmetric system.

AFFECTIVE OBJECTIVE: Willingly receive the lecture on angular momentum and rotational coupling.

AFFECTIVE SAMPLES OF BEHAVIOR:

1. Voluntarily takes notes during the lesson on angular momentum and rotational coupling.
2. Asks questions of the lecturer on angular momentum and rotational coupling.
3. Encourages others to listen attentively during the lecture on angular momentum and rotational coupling.

PART II

ORGANIZATIONAL PATTERN: Topical

STRATEGY: This lecture is designed for a presentation time of one period. Since the students are just beginning to learn astronautics the lecturer needs to determine the students' level of knowledge in basic physics. The topical method of presenting the material offers the students the opportunity to understand the definition of angular momentum and torque prior to the discussion of conservation of angular momentum and rotational coupling. Four video demonstrations are included in the presentation showing how the material applies to satellite motion. A question and answer slide follows each section to further determine student understanding of the material.

LESSON OUTLINE

MP 1. Angular momentum defined.

a. Newton's Second Law

b. Cross product of position vector with F

c. Equation for angular momentum

d. The total torque equation

MP 2. Newton's Third Law.

a. Internal forces

b. Total torque and internal forces

c. Isolated system

d. Conservation of angular momentum

MP 3. Rotational coupling.

a. Euler's rotational equations of motion

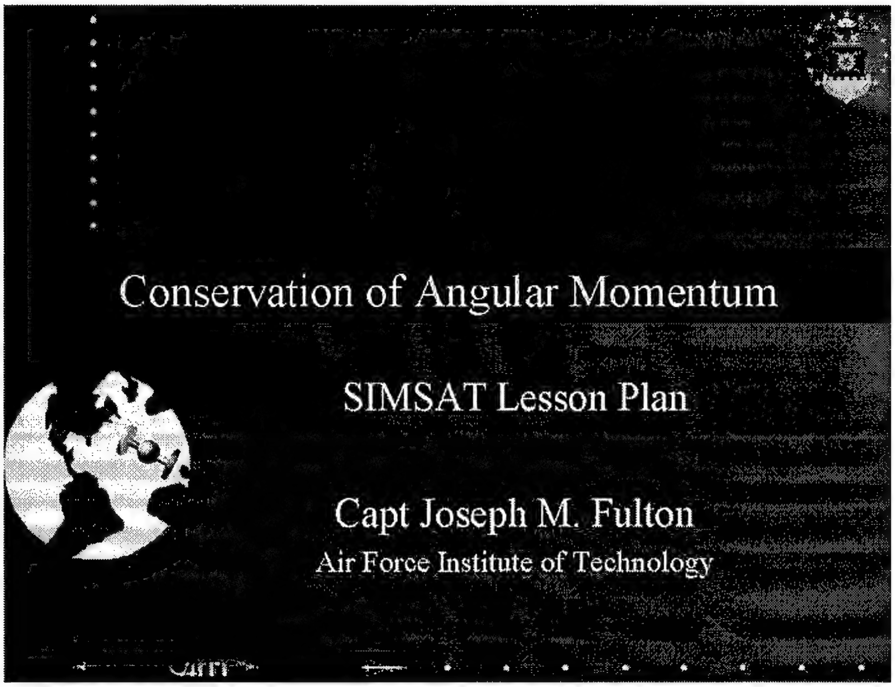
b. Rotating about three axes

c. Isolated system with an angular velocity

d. Axisymmetric systems

D.2 Lesson Plan Slides

The researcher used Microsoft PowerPoint to create the lesson plan slides. These slide are located on the ground station PC in the file c:/Fulton/briefings/lesson1.ppt. A printout of the lesson plan slide are shown in the following figures:



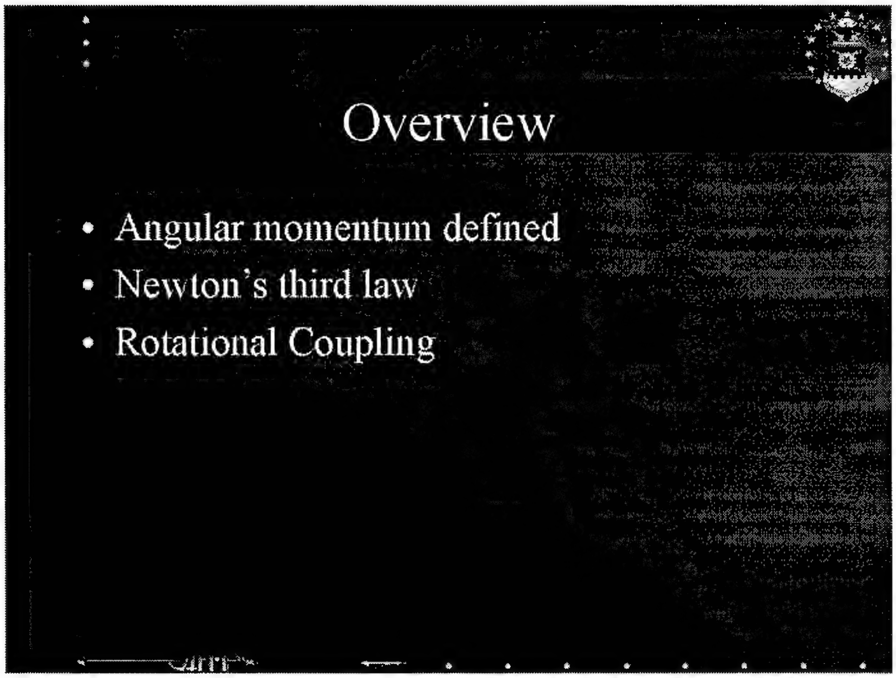
Conservation of Angular Momentum

SIMSAT Lesson Plan

Capt Joseph M. Fulton

Air Force Institute of Technology

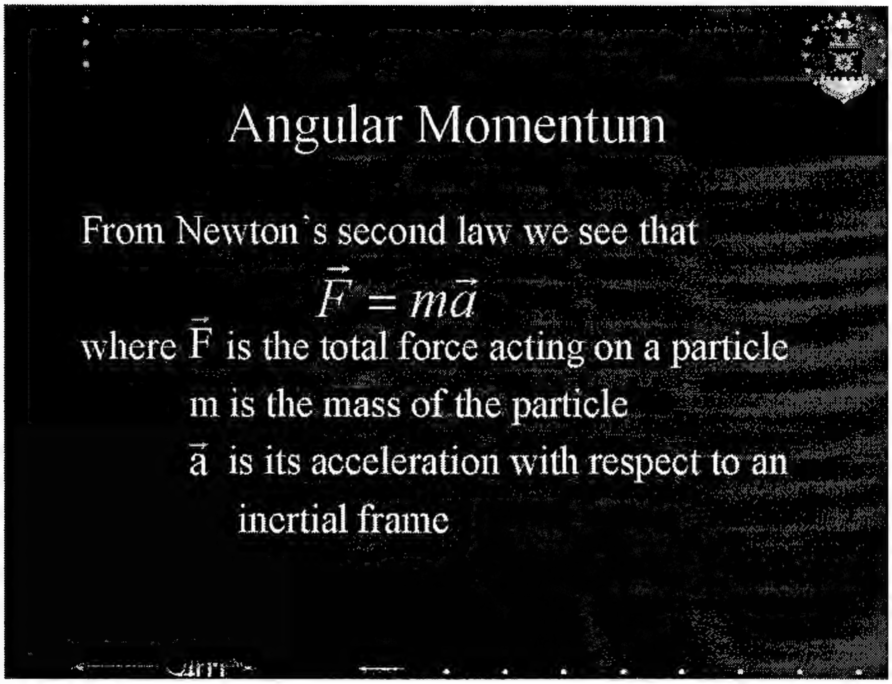
Figure D.1 Angular Momentum Lesson Plan–Slide 1



Overview

- Angular momentum defined
- Newton's third law
- Rotational Coupling

Figure D.2 Angular Momentum Lesson Plan–Slide 2



Angular Momentum

From Newton's second law we see that

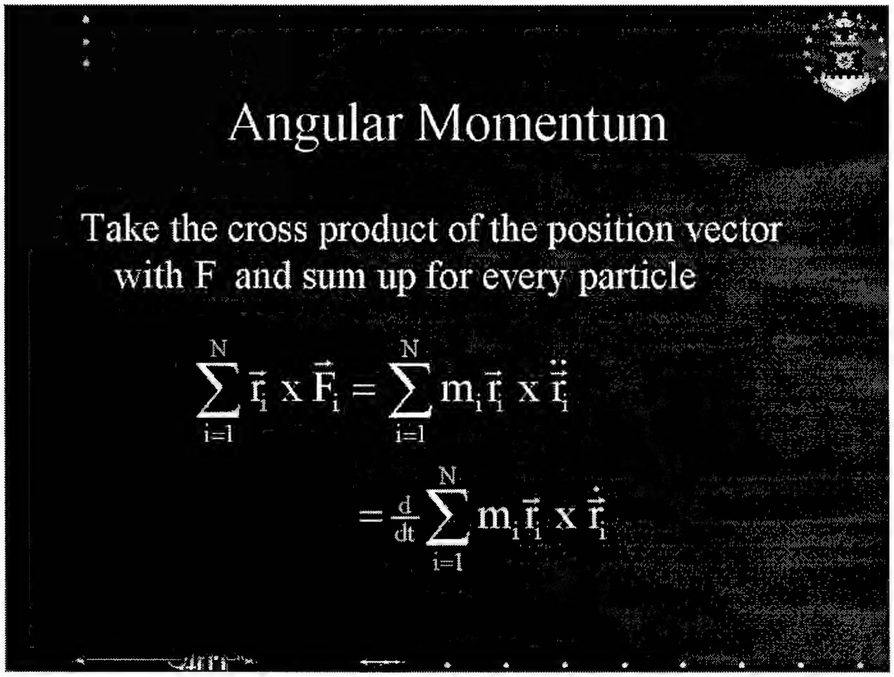
$$\vec{F} = m\vec{a}$$

where $\vec{F}$ is the total force acting on a particle

m is the mass of the particle

$\vec{a}$ is its acceleration with respect to an incrtial frame

Figure D.3 Angular Momentum Lesson Plan–Slide 3



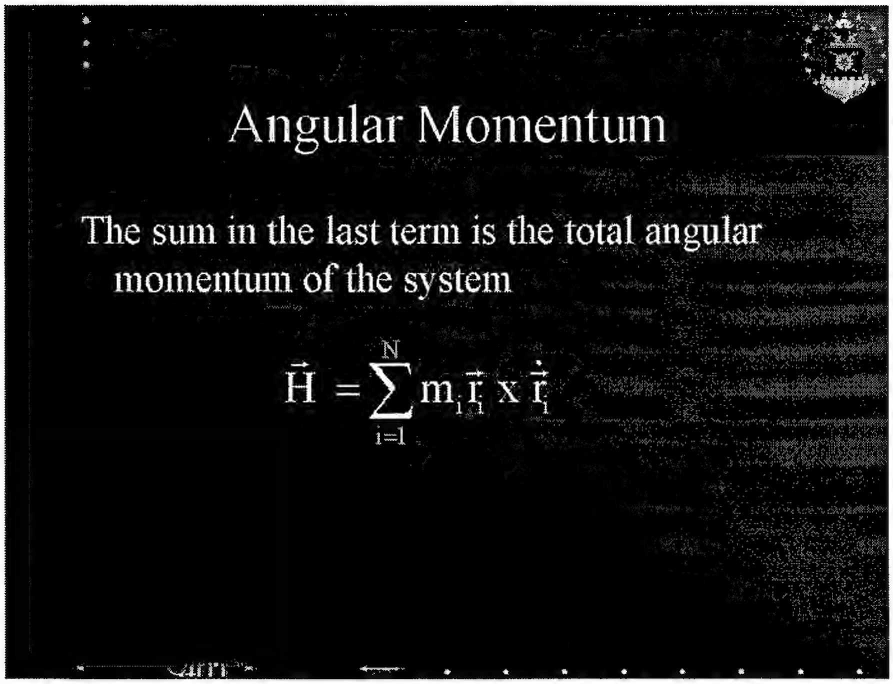
Angular Momentum

Take the cross product of the position vector with F and sum up for every particle

$$\sum_{i=1}^{N} \vec{r}_i \times \vec{F}_i = \sum_{i=1}^{N} m_i \vec{r}_i \times \ddot{\vec{r}}_i$$

$$= \frac{d}{dt} \sum_{i=1}^{N} m_i \vec{r}_i \times \dot{\vec{r}}_i$$

Figure D.4 Angular Momentum Lesson Plan–Slide 4

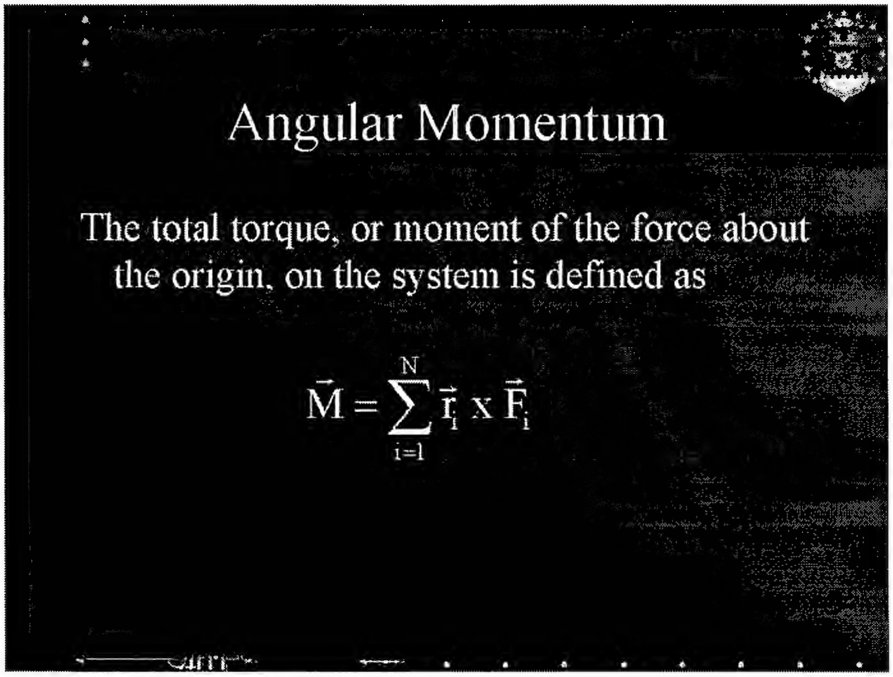


Angular Momentum

The sum in the last term is the total angular momentum of the system

$$\vec{H} = \sum_{i=1}^{N} m_i \vec{r}_i \text{ x } \dot{\vec{r}}_i$$

Figure D.5 Angular Momentum Lesson Plan–Slide 5

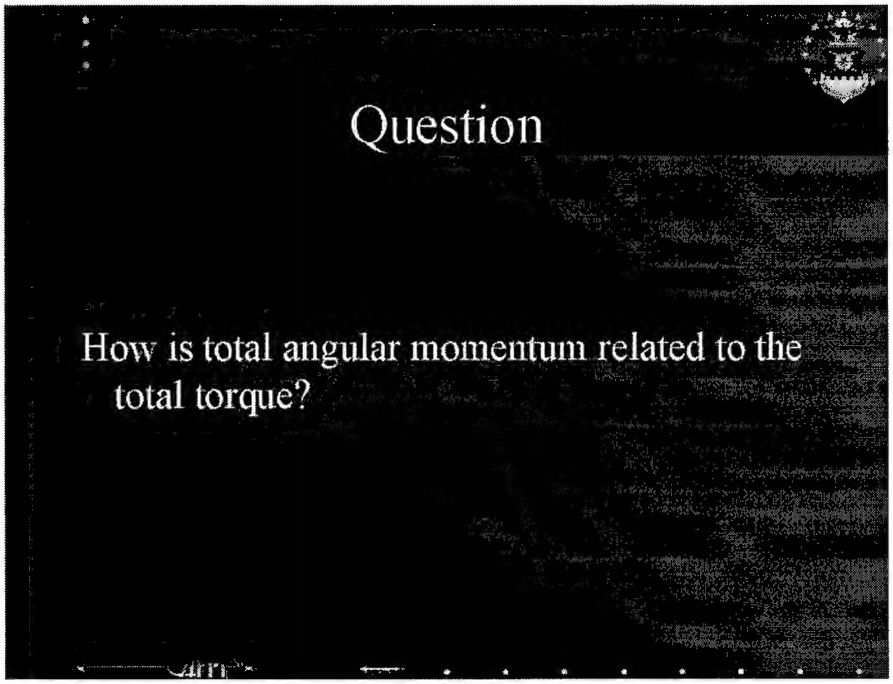


Angular Momentum

The total torque, or moment of the force about the origin, on the system is defined as

$$\vec{M} = \sum_{i=1}^{N} \vec{r}_i \text{ x } \vec{F}_i$$

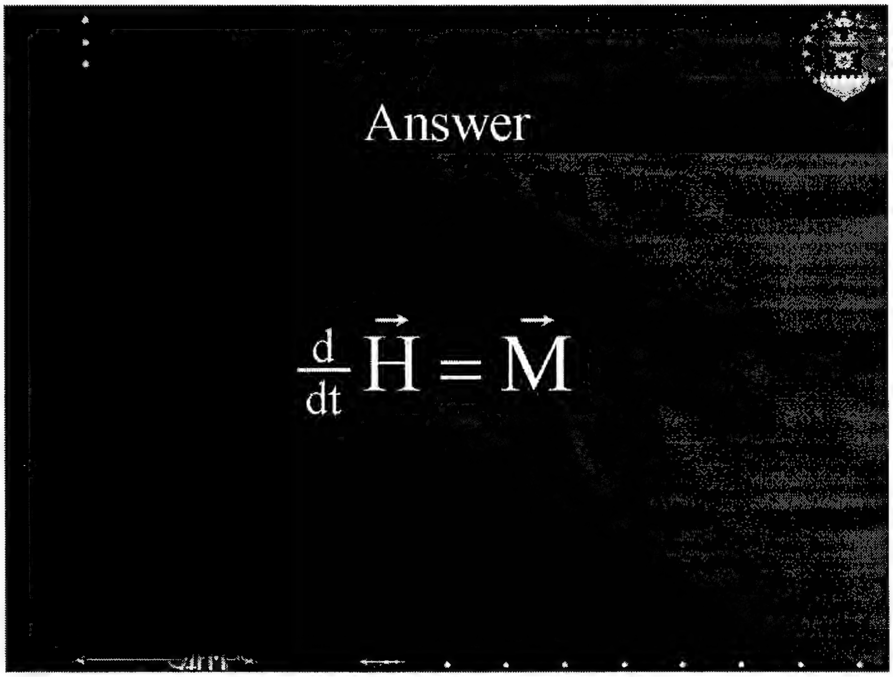
Figure D.6 Angular Momentum Lesson Plan–Slide 6



Question

How is total angular momentum related to the total torque?

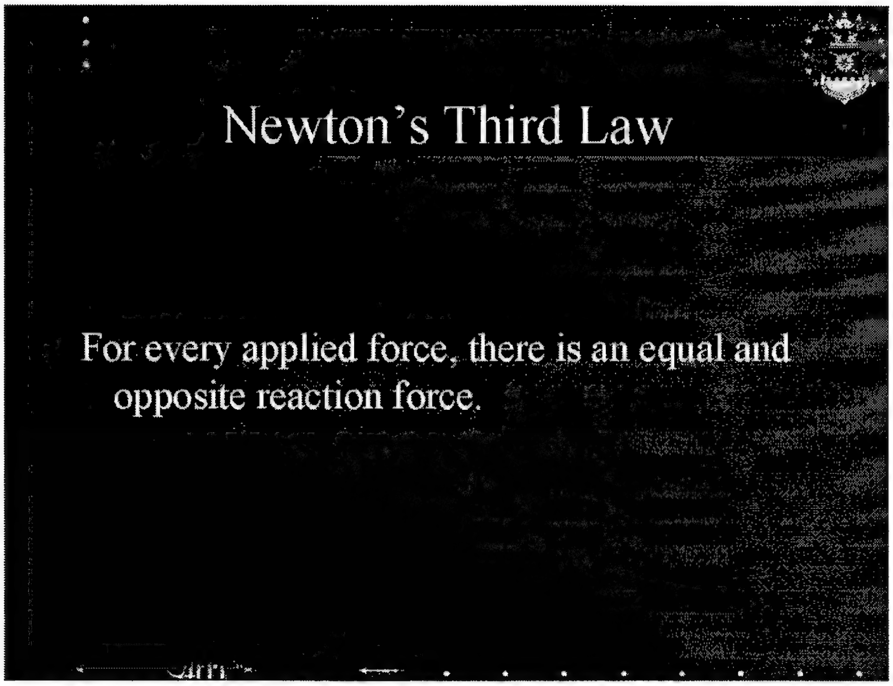
Figure D.7 Angular Momentum Lesson Plan–Slide 7



Answer

$$\frac{d}{dt}\vec{H} = \vec{M}$$

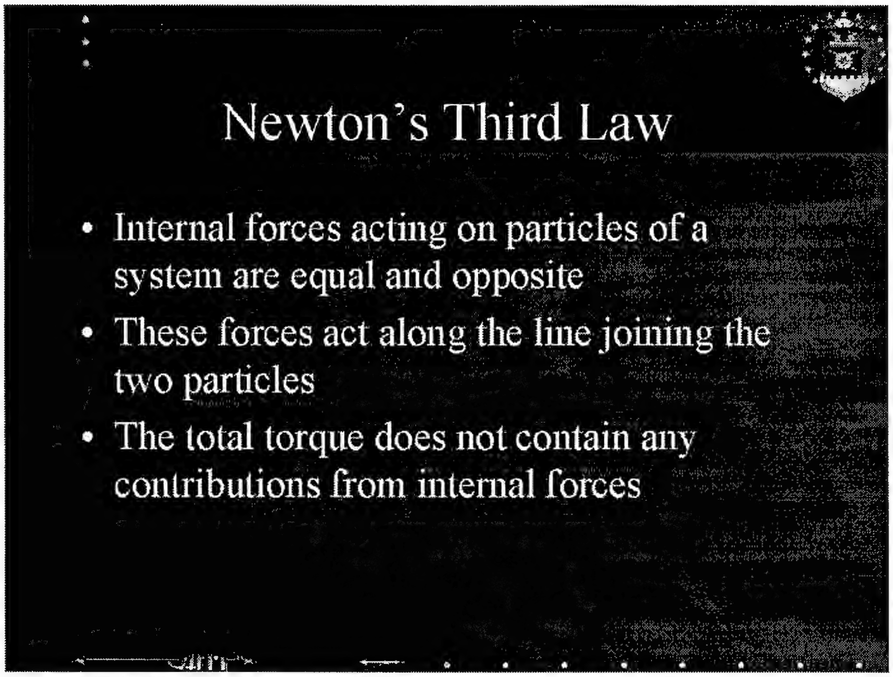
Figure D.8 Angular Momentum Lesson Plan–Slide 8



Newton’s Third Law

For every applied force, there is an equal and opposite reaction force.

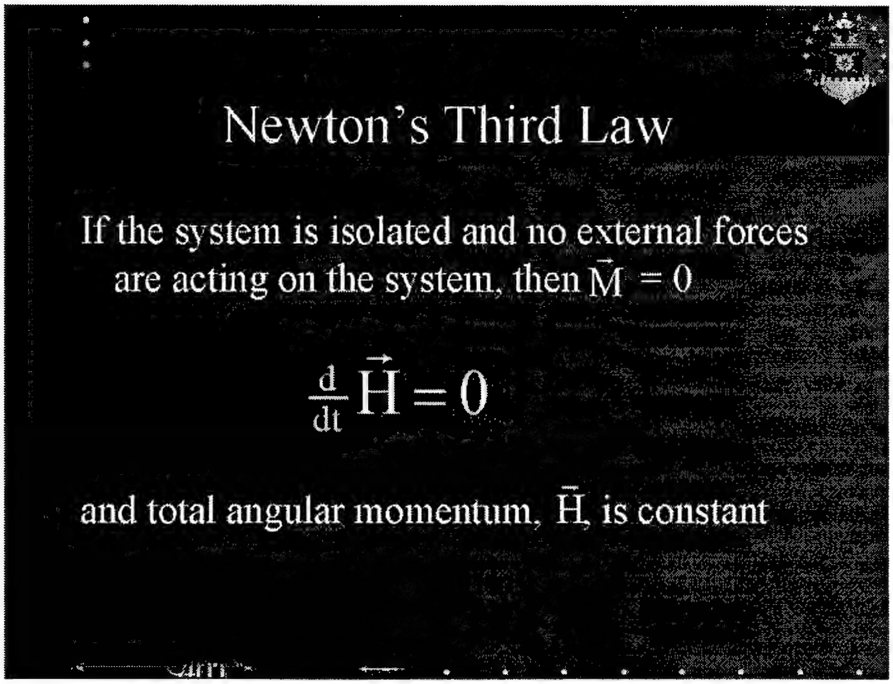
Figure D.9 Angular Momentum Lesson Plan–Slide 9



Newton’s Third Law

- Internal forces acting on particles of a system are equal and opposite
- These forces act along the line joining the two particles
- The total torque does not contain any contributions from internal forces

Figure D.10 Angular Momentum Lesson Plan–Slide 10



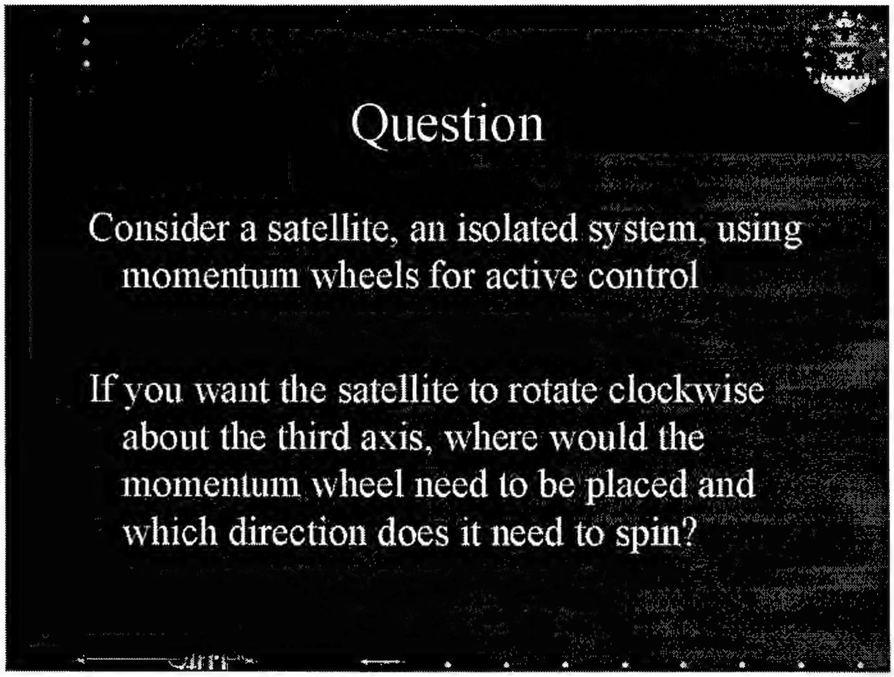
Newton's Third Law

If the system is isolated and no external forces are acting on the system, then $\vec{M} = 0$

$$\frac{d}{dt}\vec{H} = 0$$

and total angular momentum, $\bar{H}$, is constant

Figure D.11 Angular Momentum Lesson Plan–Slide 11

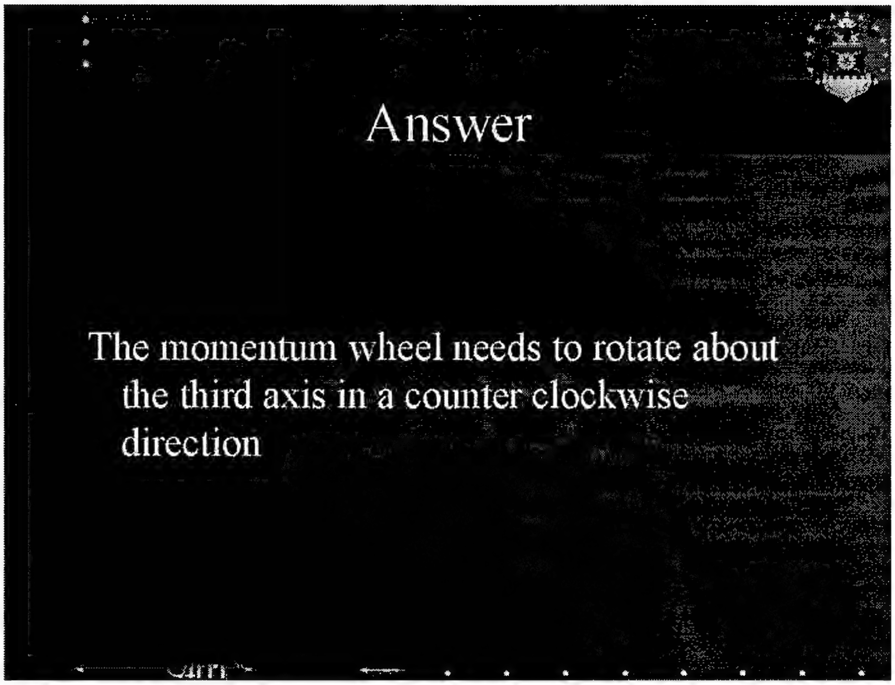


Question

Consider a satellite, an isolated system, using momentum wheels for active control

If you want the satellite to rotate clockwise about the third axis, where would the momentum wheel need to be placed and which direction does it need to spin?

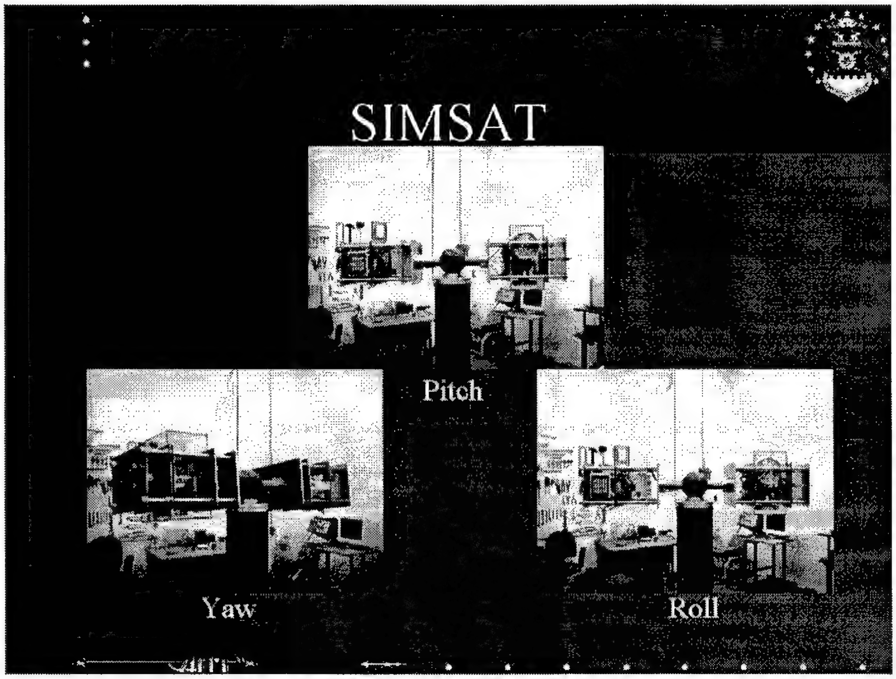
Figure D.12 Angular Momentum Lesson Plan–Slide 12



Answer

The momentum wheel needs to rotate about the third axis in a counter clockwise direction

Figure D.13 Angular Momentum Lesson Plan–Slide 13



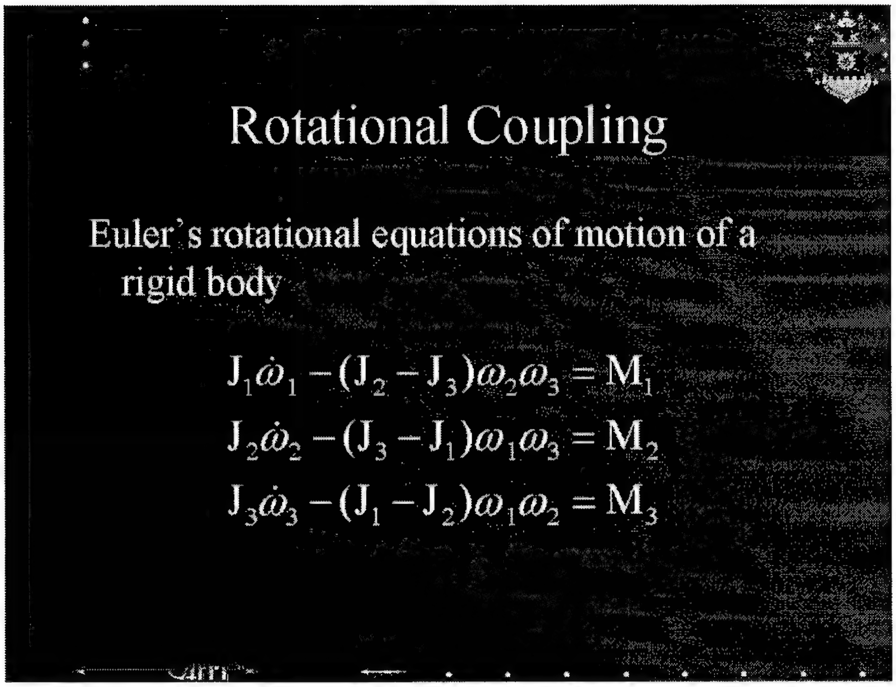
SIMSAT

Pitch

Yaw

Roll

Figure D.14 Angular Momentum Lesson Plan–Slide 14

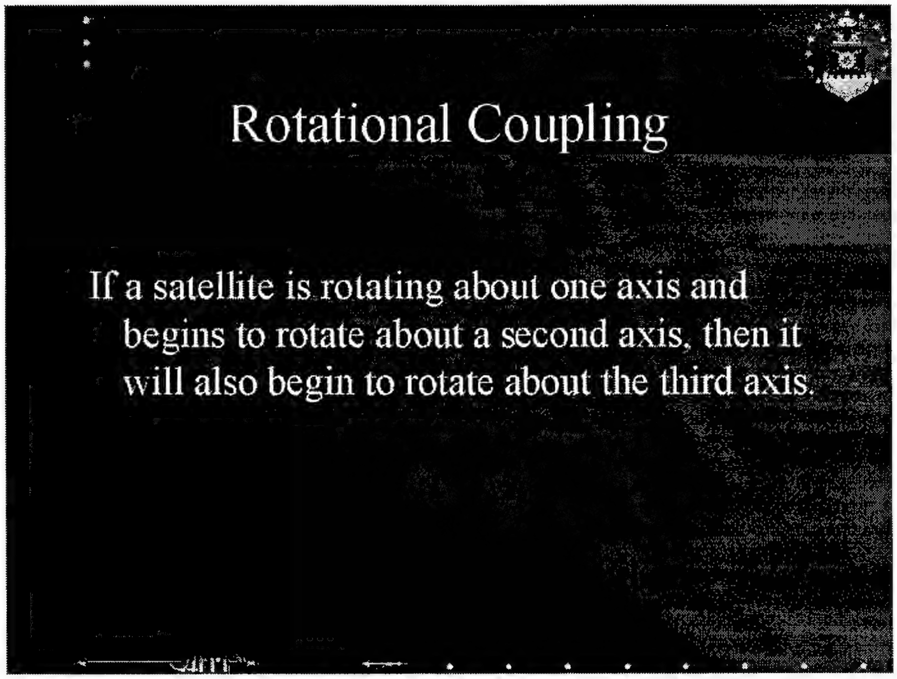


Rotational Coupling

Euler's rotational equations of motion of a rigid body

$$J_1\dot{\omega}_1 - (J_2 - J_3)\omega_2\omega_3 = M_1$$
$$J_2\dot{\omega}_2 - (J_3 - J_1)\omega_1\omega_3 = M_2$$
$$J_3\dot{\omega}_3 - (J_1 - J_2)\omega_1\omega_2 = M_3$$

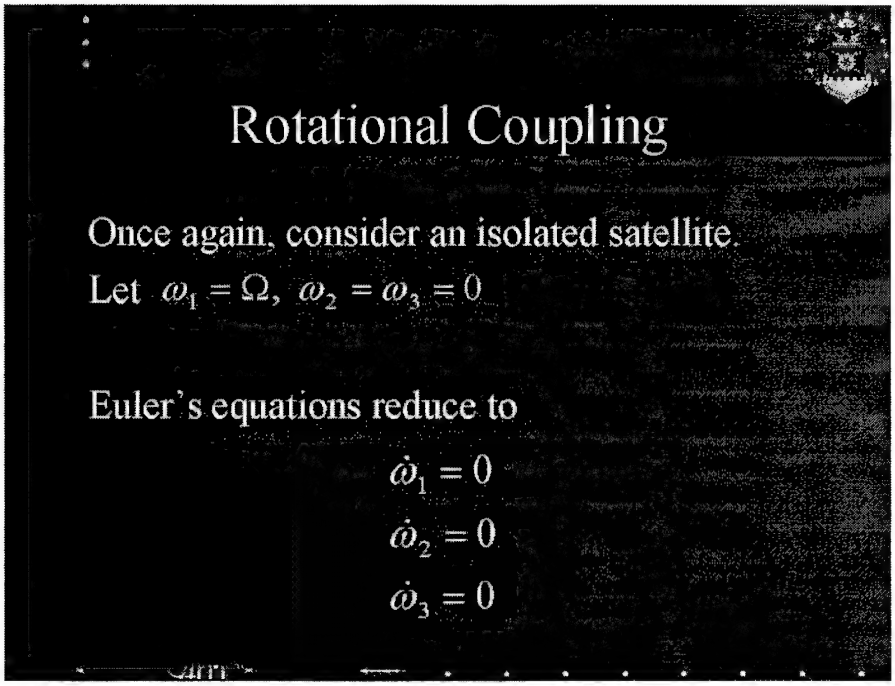
Figure D.15 Angular Momentum Lesson Plan–Slide 15



Rotational Coupling

If a satellite is rotating about one axis and begins to rotate about a second axis, then it will also begin to rotate about the third axis.

Figure D.16 Angular Momentum Lesson Plan–Slide 16



Rotational Coupling

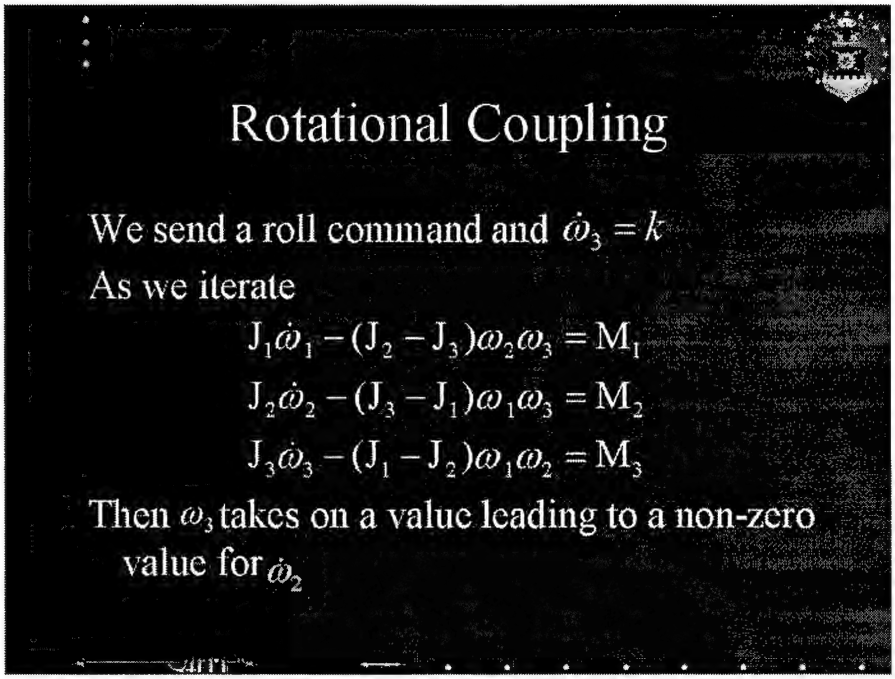
Once again, consider an isolated satellite.

Let $\omega_1 = \Omega,\ \omega_2 = \omega_3 = 0$

Euler's equations reduce to

$$\dot{\omega}_1 = 0$$
$$\dot{\omega}_2 = 0$$
$$\dot{\omega}_3 = 0$$

Figure D.17 Angular Momentum Lesson Plan–Slide 17



Rotational Coupling

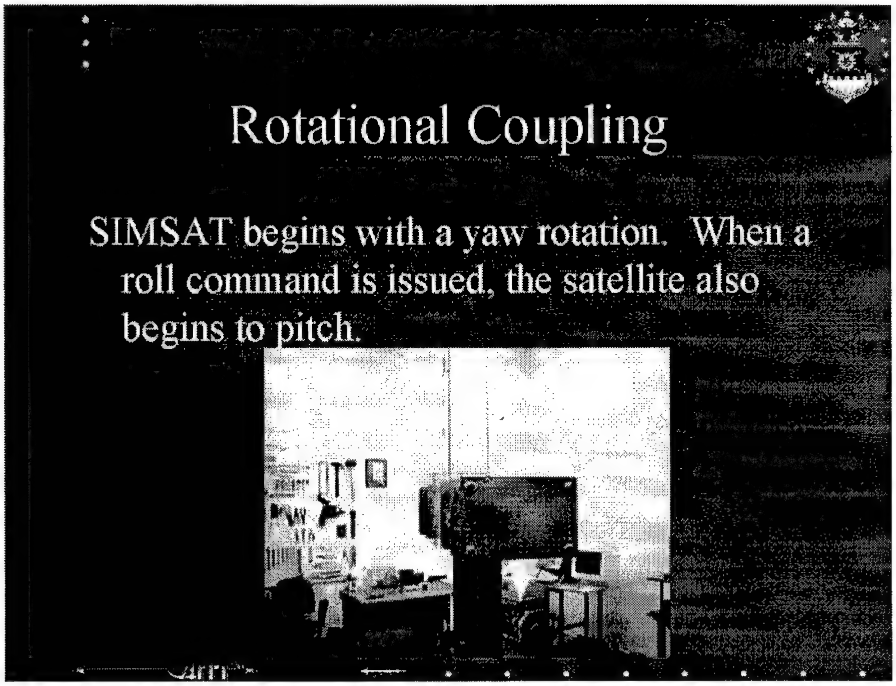
We send a roll command and $\dot{\omega}_3 = k$

As we iterate

$$J_1\dot{\omega}_1 - (J_2 - J_3)\omega_2\omega_3 = M_1$$
$$J_2\dot{\omega}_2 - (J_3 - J_1)\omega_1\omega_3 = M_2$$
$$J_3\dot{\omega}_3 - (J_1 - J_2)\omega_1\omega_2 = M_3$$

Then ω_3 takes on a value leading to a non-zero value for $\dot{\omega}_2$

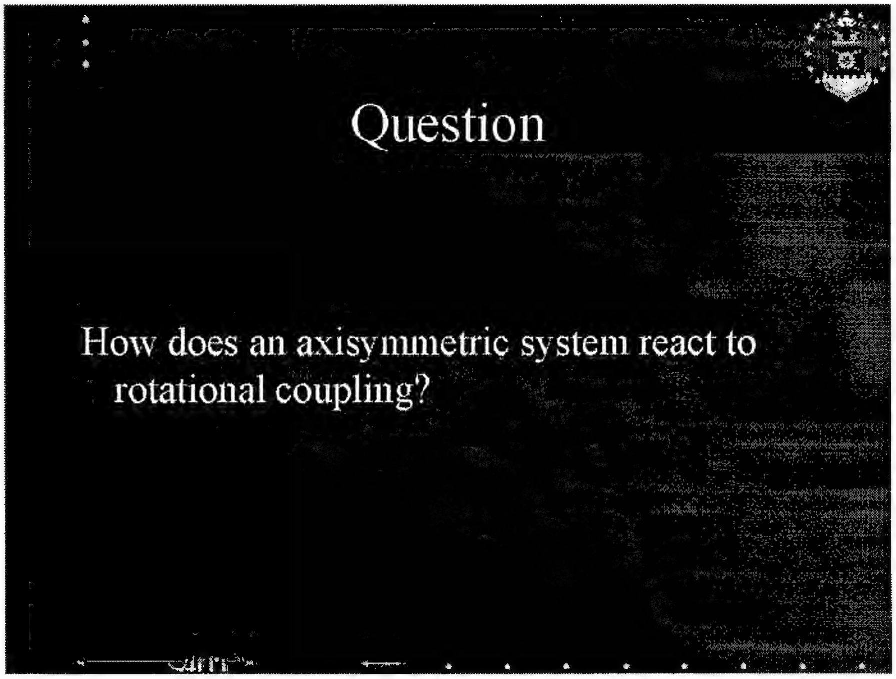
Figure D.18 Angular Momentum Lesson Plan–Slide 18



Rotational Coupling

SIMSAT begins with a yaw rotation. When a roll command is issued, the satellite also begins to pitch.

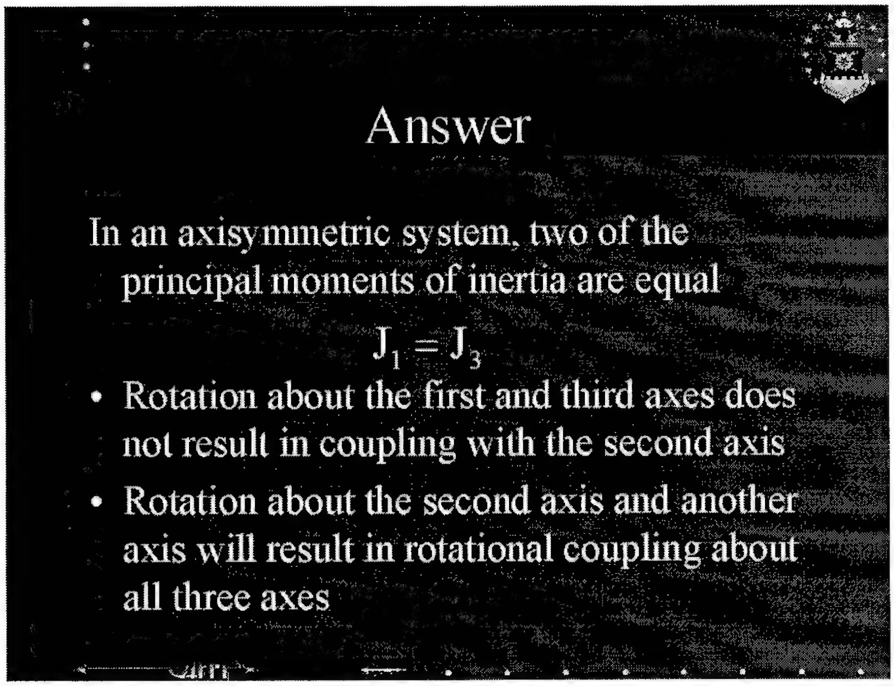
Figure D.19 Angular Momentum Lesson Plan–Slide 19



Question

How does an axisymmetric system react to rotational coupling?

Figure D.20 Angular Momentum Lesson Plan–Slide 20



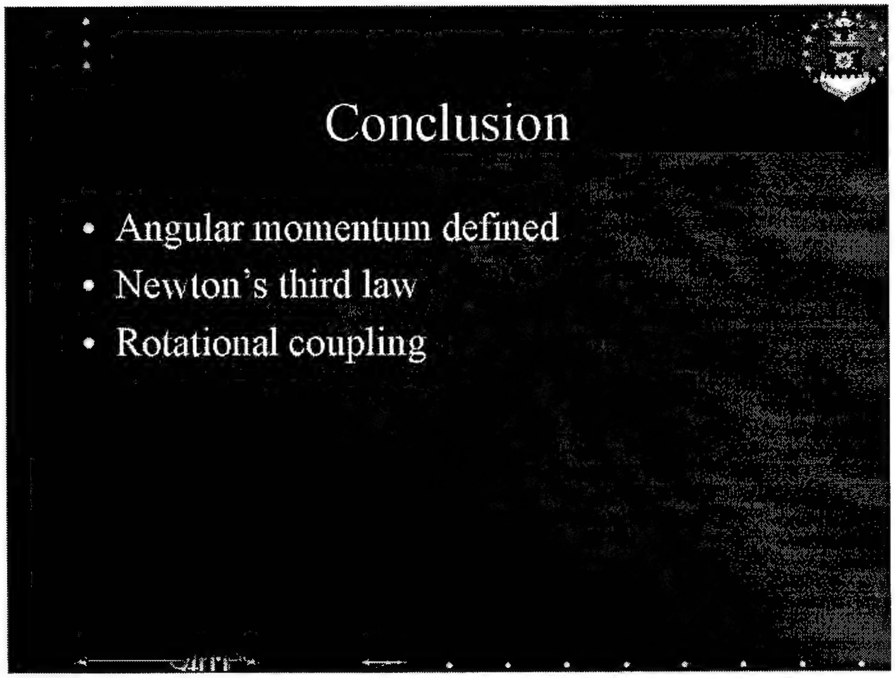
Answer

In an axisymmetric system, two of the principal moments of inertia are equal

$$J_1 = J_3$$

- Rotation about the first and third axes does not result in coupling with the second axis
- Rotation about the second axis and another axis will result in rotational coupling about all three axes

Figure D.21 Angular Momentum Lesson Plan–Slide 21



Conclusion

- Angular momentum defined
- Newton's third law
- Rotational coupling

Figure D.22 Angular Momentum Lesson Plan–Slide 22

D.3 Laboratory Experiment

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright Patterson Air Force Base, Ohio

EXPERIMENTAL OBJECTIVE: Students may use *SIMSAT* to demonstrate angular momentum concepts learned in formal lecture.

EXPERIMENTAL OVERVIEW: *SIMSAT* is located in building 640, room 146. Two students and the instructor are required to conduct this experiment. One student will input telemetry commands on the ground station PC while the other student will stabilize the system in between each maneuver. The instructor is present to provide guidance and for safety concerns.

The following procedure is used to operate *SIMSAT*.

1. Turn the power switch on the air compressor to Auto
2. Locate the rubber glove, denatured alcohol, and tissue in the third drawer of the desk
3. While wearing the rubber glove, remove the cover from the air bearing cup
4. Clean the surface of the air bearing cup using the tissue and alcohol
5. Remove the protective case from *SIMSAT*'s spherical rotor
6. Clean the spherical rotor's surface with the tissue and alcohol
7. Replace the rubber glove and alcohol back in the drawer
8. Locate the hydraulic crane and lifting straps
9. Secure the lifting straps to the hollow shaft on either side of the spherical rotor
10. Attach the lifting straps to the hook at the end of the hydraulic crane
11. Remove the anti-tip collars from the support stanchions
12. Slowly raise *SIMSAT* off of the support stanchions using the hydraulic crane
13. Move the support cart off to one side

14. Raise *SIMSAT* to a height greater than the top of the air pedestal
15. Carefully maneuver the hydraulic crane to place the spherical rotor precisely above the air bearing cup
16. Gently lower *SIMSAT* onto the air bearing cup while another student guides the spherical rotor into position
17. While one student supports *SIMSAT*, remove the lifting straps from the hollow shaft
18. Move the hydraulic crane off to one side
19. Balance the system using the fine-tuning counterweight mechanism located at one end of *SIMSAT*
20. Turn on the 12V and 24V power switches on *SIMSAT*
21. Caution – do not turn on the 36V power switch at this time
22. *SIMSAT* will begin to rotate about the vertical axis as a result of the gyro spinning up
23. Turn on the ground station PC
24. Insert the SIMSAT 2000 Zip disk into the removable disk drive
25. Select the RadioLAN connection outlined in Section 4.2
26. Double click on the MATLAB 5.3 icon
27. Ensure MATLAB successfully connects to dSPACE
28. In MATLAB's command window type simulink and press enter
29. In the Simulink Library Browser select the Open a Model icon
30. Select the file D:/SIMSAT2000/simtestopenloop.mdl
31. Download this model to the AutoBox using the procedures outlined in Section 4.3.4
32. Caution – if the 36V power switch on *SIMSAT* is turned on, the momentum wheels may begin to rotate
33. On the ground station PC desktop, double click on the dSPACE ControlDesk icon
34. Ensure dSPACE successfully connects to the AutoBox (refer to Section 4.4

35. Select File from Control Desk's main menu
36. Select Open Experiment
37. Select the file C:/simsatmodel/simtest/simtest.cdx
38. Select the Stop RTP icon from the top menu
39. Turn on the 36V power switch on *SIMSAT*
40. Select the Start RTP icon from the top menu of Control Desk
41. Select the Animation Mode icon from the top menu

Now, *SIMSAT* is ready to receive commands from the ground station PC. Maneuver commands are inserted in the wheel speed incremental input windows of Control Desk. The students may command each wheel separately and watch *SIMSAT* respond to the command. Rotational coupling is demonstrated by allowing *SIMSAT* to rotate about the vertical axis and then inputting a roll command. For safety concerns, limit commands to $\pm 1V$.

To secure *SIMSAT* once the experiment is complete, accomplish the following:

1. On the Control Desk, select the Stop RTP icon from the top menu
2. Exit Control Desk
3. Exit MATLAB
4. Turn off the ground station PC
5. Turn off the 36V, 24V, and 12V power switches on *SIMSAT*
6. Position the hook of the hydraulic crane above the spherical rotor
7. Attach the lifting straps to the hollow shaft on either side of the spherical rotor
8. Gently raise the crane to lift *SIMSAT* off of the air bearing cup
9. Move the hydraulic crane away from the air pedestal
10. Position the support cart below *SIMSAT*
11. Lower *SIMSAT* onto the support stanchions

12. Remove the lifting straps
13. Place the protective casing around the spherical rotor
14. Secure the anti-tip collars to the support stanchions
15. Replace the metal cover on the air bearing cup
16. Turn the air compressor power switch to Off
17. Recharge the batteries on *SIMSAT*

Bibliography

1. Agnes, Gregory. “MECH533: Intermediate Space Flight Dynamics Class Notes.” Air Force Institute of Technology, Spring 1998.
2. Ambron, SueAnn. *The Interactive Learning Revolution: Multimedia in Education and Training*. New York: Nichols Publishing, 1990.
3. Borsook, T. K. and N. Higginbotham-Wheat. “Interactivity: What is it and What Can it Do for Computer-Based Instruction?,” *Educational Technology*, *vol. 31*(no. 10):11–17 (1991).
4. Bowers, C. A. *The Cultural Dimensions of Educational Computing*. New York: Teachers College Press, 1988.
5. Bruner, J. S. *Studies in Cognitive Growth*. New York: John Wiley and Sons, 1966.
6. Chobotov, Vladimir A. *Spacecraft Attitude Dynamics and Control*. Malabar, Florida: Krieger Publishing Company, 1991.
7. Colebank, James and others. *SIMSAT: A Satellite System Simulator and Experimental Test Bed for Air Force Research*. MS thesis, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, March 1999.
8. Collins, Janet and others. *Teaching and Learning with Multimedia*. London: Routledge, 1997.
9. Elachi, Charles. *Introduction to the Physics and Techniques of Remote Sensing*. New York: John Wiley and Sons, 1987.
10. Fifield, S. and R. Peifer. “Enhancing Lecture Presentations in Introductory Biology with Computer-Based Multimedia,” *Journal of College Science Teaching*, *vol. 23*(no. 4):235–239 (1994).
11. Fox, Jackie. “When Worlds Collide: Demystifying Multimedia,” *PC Today*, 6 (June 1991).
12. Gagne, R. M. *The Conditions of Learning* (Third Edition). New York: Holt, Rinehart, and Winston, 1977.
13. Hanke, Michael P. *Design of the Computer Subsystem for the AFIT Simulation Satellite (SIMSAT)*. MS thesis, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1998.
14. Johnson, R. T. and others. “Effects of Cooperative, Competitive, and Individualistic Goal Structures on Computer-Assisted Instruction,” *Journal of Educational Psychology*, *vol. 77*(no. 6):668–677 (1985).
15. Kellner, Mark. “Multimedia's Mounting Momentum,” *InfoWorld*, 58 (October 21 1991).
16. Kirakowski, J. *The Digital University: Reinventing the Academy*. London: Springer-Verlag London Limited, 1998.

17. Lee, M. W. "The Match: Learning Styles of Black Children and Microcomputer Programming," *Journal of Negro Education*, *vol. 55*(no. 1):78–90 (1986).

18. Liu, David and Linda Beamer. "Multimedia as a Teaching Tool in Business Communication Course Delivery," *Business Communication Quarterly*, *vol. 60*(no. 2):51–66 (June 1997).

19. Lookatch, Richard P. "Multimedia Improves Learning–Apples, Oranges and the Type I Error," *Contemporary Education*, *vol. 68*(no. 2):110–113 (Winter 1997).

20. Maddux, Cleborne D. and others. *Educational Computing: Learning with Tomorrow's Technologies* (Second Edition). Boston: Allyn and Bacon, 1997.

21. Magel, Mark. "The Many Faces of Multimedia," *AV Video*, 68 (September 1990).

22. Merrill, Paul F. and others. *Computers in Education* (Third Edition). Boston: Allyn and Bacon, 1996.

23. Morris, Ronald L. *Computers in Education* (Seventh Edition). Guilford CT: Dushkin Publishing Group/Brown and Benchmark, 1996.

24. Myers, D. K. "Interactive Video: A Chance to Plug the Literacy Leak," *Industry Week*, 15–18 (April 1990).

25. Pelgrum, P. J. and T. Plomp. "The Worldwide Use of Computers: A Description of Main Trends," *Computers Education*, *vol. 20*:323–332 (1993).

26. Pence, H. "A Report from the Barricades of the Multimedia Revolution," *Educational Technology System*, *vol. 24*:159–164 (1996).

27. Picciano, Anthony G. *Educational Leadership and Planning for Technology* (Second Edition). Upper Saddle River NJ: Prentice-Hall, Inc., 1998.

28. Powers, Parris. "One Path to Using Multimedia in Chemistry Courses: Enlivening Student's Learning through Visual Presentations," *Journal of College Science Teaching*, *vol. 27*(no. 5):317–318 (March-April 1998).

29. Space Electronics, Inc., 81 Fuller Way, Berlin, CT 06037-1540. *Tri-axis Gas Bearing Model 2630 Instruction Manual*, 1997.

30. Sutton, R. E. "Equity and Computers in the Schools: A Decade of Research," *Review of Educational Research*, *vol. 61*(no. 4):475–503 (1991).

31. Tapscott, D. *The Digital Economy: Promise and Peril in the Age of Networked Intelligence*. New York: McGraw-Hill, 1996.

32. United States Air Force. *Guidebook for Air Force Instructors* (Afm 36-2236 Edition), September 1994.

33. Weiner, H. "Enhancing Student Performance in the Social Studies Through the Use of Multimedia Instruction." U.S. Department of Education. (ERIC Document of Reproduction Service No. ED 383 598).

34. Wie, Bong. *Space Vehicle Dynamics and Control*. Reston, Virginia: American Institute of Aeronautics and Astronautics, Inc., 1998.

35. Wiesel, William E. *Spaceflight Dynamics* (Second Edition). Boston: Irwin/McGraw-Hill, 1997.

36. Wilcox, Kimerly J. and Murray S. Jensen. "Computer Use in the Science Classroom: Proceed with Caution!," *Journal of College Science Teaching, vol. 26*(no. 4):258–264 (February 1997).

37. Zehr, Mary Ann. "Not by the Book." n. pag. http://www.edweek.org/ew/vol-18/14tech.h18, June 1999.

Vita

Captain Joseph M. Fulton was born on 5 September 1969 in Ann Arbor, Michigan. He graduated fifth in his class from Roscommon High School in Roscommon, Michigan in June 1987. He entered undergraduate studies at Michigan State University in East Lansing, Michigan and graduated with a Bachelor of Science degree in Astrophysics in June 1991. He earned his Air Force commission while attending Officer Training School in May 1993.

Captain Fulton began Undergraduate Missile Training with the 4315th Combat Crew Training Squadron, Vandenberg AFB, CA, in September 1993. He completed training as a Distinguished Graduate in February 1994 and was assigned to the 10th Missile Squadron, Malmstrom AFB, MT. During the five years of service at Malmstrom he served in a variety of positions including Deputy Combat Crew Commander, Flight Deputy Commander, Instructor/Evaluator Deputy Combat Crew Commander, Missile Combat Crew Commander, and Squadron Command Post Flight Commander. Prior to his leaving the 10th Missile Squadron he completed his Master's in Business Administration in Personal Finance from City University, Bellevue, WA in May 1998.

In August 1998, he began his Master's degree in Astronautical Engineering at the School of Engineering, Air Force Institute of Technology (AFIT). While studying at AFIT he was inducted into Sigma Gamma Tau and elected Vice President of the local chapter. His follow-on assignment is to the Astronautics Department of the United States Air Force Academy as an instructor.

He is married to the former Maryann M. Gibala of Pinconning, MI, and together have two children, Jenna and Emily.

Permanent address: 7960 Camfield Circle
Colorado Springs, CO 80920
e-mail: gilthane@worldnet.att.net

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 09 MAR 00	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From - To)*

4. TITLE AND SUBTITLE
ATTITUDE CONTROL AND MULTIMEDIA REPRESENTATION OF AIR FORCE INSTITUTE OF TECHNOLOGY'S (AFIT'S) SIMULATION SATELLITE (SIMSAT)

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Joseph M. Fulton, Capt, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology
2950 P Street, Bldg 640
Wright-Patterson AFB, OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GA/ENY/00M-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFIT/ENY
2950 P Street, Bldg 640
Wright-Patterson AFB, OH 45433-7765

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; Distribution unlimited

13. SUPPLEMENTARY NOTES
Advisor: Gregory S. Agnes, Capt, USAF
DSN 785-6565 ext. 4317 COM (937) 255-6565 ext. 4317

14. ABSTRACT
This document describes the systematic construction of a laboratory-based satellite simulator. The simulation satellite (SIMSAT) system will provide a useful tool for resident staff while teaching attitude control concepts. A brief overview of attitude control theory is provided as well as a discussion of the benefits of multimedia use in education. A detailed discussion of the satellite's components allows the reader to become familiar with each piece of SIMSAT. Software control models are provided as well as a multimedia lesson plan on satellite attitude control. Also included in this document are potential experimental uses in the areas of attitude control, precision pointing, and vibration suppression as well as continued modification of the multimedia presentation capabilities.

15. SUBJECT TERMS
Angular Momentum, Attitude Control, Momentum Wheels, Multimedia Education, PD Controller, Rigid Body Dynamics, Rotational Kinematics ,Simulation Satellite, Torque, Wireless Communication

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Gregory S. Agnes, Capt, USAF
a. REPORT UNCLASS	**b. ABSTRACT** UNCLASS	**c. THIS PAGE** UNCLASS	UL	253	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-3069